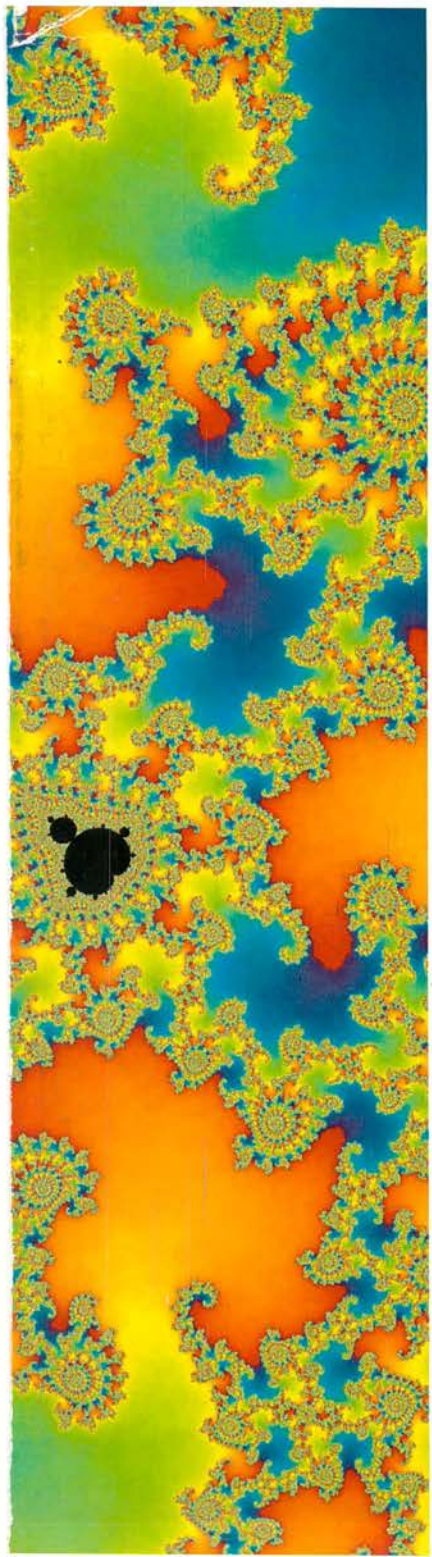


OCTOBER 2012

Vol. 23 N° 5 £4.50

Archive

The subscription magazine for users of the Acorn Archimedes and its successors

Newsbriefs ... Jim Nagel ... 3

First taste of Raspberry Pi ... Moss Rolfe ... 4
Alpha Risc os on the Pi ... Chris Hall ... 5
Raspberry Pi at Rougol ... Bernard Boase ... 8

The Risc os shows of 2012
Wakefield ... Ron Briscoe ... 10
Southwest ... John Peachey ... 14
Midlands, Netherlands

Charm – a practical guide ... Peter Nowosad ... 18

Iyonix power-supply warning John Ballance ... 21

A different viewpoint ... Gerald Fitton ... 22

Reading problems *(Sound 32)* ... Jim Lesurf ... 32

Free to share: PD column ... Steve Fryatt ... 36

Mac Matters ... Mark Stephens ... 41

Numbers game *(6)* ... Keith Hopper ... 43

Compiling with GCC *(20)* ... Rob Johnson ... 48

Secret life of algorithms *(7)* Nicholas Cutler ... 54
Square roots

PC Bits: first look at Winadows 8 Bimal Jangra ... 60
Fine for a touchscreen, not for a desktop

Archive files to download ... 2
Risc os contacts ... 59
Kit needing new homes ... 4, 30, 35

hint&tips
Finding a lost file in an over-full folder ... 17
Noisy RiscPC fan ... 31
Redirecting output ... 42

This copy of Archive was printed digitally to be ready for the London show

ednotes

At long last – Here is Archive 23:5, after a lamentably long gap. If you're reading this issue hot off the press at the London show, the reports on pages 10–17 of shows earlier this year will serve as a guide to interesting stuff being demonstrated again at this show. The main buzz is definitely the Raspberry Pi. Even the education secretary seems to have taken the Pi's message on board, that children need to learn computer science like they did a generation ago, rather than "ICT" lessons on how to drive Microsoft Word.

Now that my hospital visits are finished, I'm thankful (and I don't use the word glibly) to be able to say that I had no nasty effects from either the disease or the cure. But the 37 daily trips, 1½ hours each way plus waiting – for a ten-second zap – plus tiredness afterwards, certainly did devour my life for two or three months. So many unfinished jobs and clutter piled up that even now, when I no longer have an excuse for being lazy, just looking at the piles makes me tired. But we're getting there.

Best wishes to all. Hope to see many of you at the show in London. *—jim*

Cover – *Another Mandelbrot rendered by the Charm programming language (see page 17).*

• Archive was founded in 1987 in support of the Acorn Archimedes computer and its successors using the RISC OS operating system. It is entirely produced with RISC OS software. Body text is 9½pt Plantin; headings are EFF Eric (Gill Sans). Wordcount this issue: 27,417.

Lithoprinted via A2B Print Solutions, Somerset

Archive Editor: **Jim Nagel** *jim@archivemag.co.uk* Founding editor: Paul Beverley

SUBSCRIPTIONS – £50 in the UK for 12 issues, £56 to the rest of Europe, £62 to rest of the world

All the past 22 years on CD or DVD: p47

Payment methods: *cheque to "Abbey Press" at address on back cover, or direct bank transfer with an email to confirm details – sortcode* **08-92-50** *(Co-op, Skelmersdale), account* **7072 6592** *(Abbey Press). Another route (less cost-effective) is via Google Checkout at www.archivemag.co.uk (Prices page).*

Archive files to download

- Mandelbrot pictures and program files in the Charm language, page 17
- Spreadsheets and other files for Gerald Fitton's column, page 22
- Application to go with Jim Lesurf's article about sound, page32
- Files illustrating Rob Johnson's article about the GCC compiler, page 48
- Square-root program to accompany Nicholas Cutler's algorithms, page 54
- **Cumulative Arcscan data: an index to your copies of Archive.**

These files are available free from our website at **archivemag.co.uk** *– click on "Files". Ifyouarenotonline and would like the files on a floppy, contact the editor.*

Also: all the web links printed in this issue are on the website in clickable form.

Newsbriefs Jim Nagel

New Arm hardware now runs classic applications

With the arrival of a new, free, version of Aemulor that enables classic software to run on new Arm hardware, R-Comp says its Armini machine "may well be the most compatible system released since the RiscPC".

Aemulor was programmed by Adrian Lees nearly a decade ago and marketed by Neil Spellings Computer Services. With it, software designed for older 26-bit Arm processors could run on the Iyonix with its 32-bit Xscale chip. The updated emulator now does the same job for the newest generation, the Arm v7 processors around which the Pandaboard, Beagleboard, Armini and Raspberry Pi are built.

That means perennial favourites such as Impression Publisher, Rhapsody and Complete Animator (and even Sibelius?) can now run on the new machines. Games like Heroes of Might and Magic work too.

"We're aware that many users had held off buying their next-generation RISC OS computer pending definitive reassurance that old software will work on the new system," said Andrew Rawnsley of R-Comp, implying that the Armini is now fit to claim the mantle of Acorn's RiscPC. "The future of RISC OS and its applications is stronger now than it has been for a long time."

Aemulor Professional for Arm Cortex A8 and A9 processors can be downloaded free from *buyit.spellings.net*. R-Comp supplies it as standard with new Armini machines.

What's this new computer?

R-Comp told Archive a week before the London show that they "hope" to have a new RISC OS computer system ready for the day. No further details were offered.

Top-of-the-line IPS display panels will be one of the star attractions on the stand, however. They boast "superb viewing angles and top-notch colour accuracy". Prices have fallen recently, and IPS screens now sell at roughly the same price as older non-IPS monitors. Resolution is typically 1920×1080 pixels.

R-Comp will debut its new Plingstore software-download service for RISC OS at the show. It's a similar idea to Apple's app store (avoiding Apple's lawyers by choosing a different name).

Juicier Raspberry Pi

The Raspberry Pi Model B now ships with 512 megabytes of ram as standard, at the same $35 price as the 256M first edition.

Drag'n'Drop has a new general

Christopher Dewhurst has stepped in to take over Drag'n'Drop, the magazine published electronically as PDF for RISC OS users and people with Acorn 8-bit computers. Paul Stewart announced in September that he would produce no more issues of the magazine he started three years ago. Christopher said he plans to continue Drag'n'Drop in the same vein for now: "a quarterly collourful downloadable publication like the ole glossy Beeb mags with games, hints and tips, programming and so on."

First taste of Raspberry Pi

Noel (Moss) Rolfe

The first thing you notice about the Pi is how small it is. I had ordered mine through Farnell (Element 14), so when it arrived in the post it had my free T-shirt with it.

Goes nicely with a cuppa

When I first unpacked it I sat it on a cardboard box in front of the TV, plugged in the cables, fired up a computer and made a cup of coffee while I waited for the Debian OS image file to download.

The next step was to download a program called Win32Diskimager to transfer the image file to the SD card. The process of burning the image file to the card turned out to be quick and painless.

Next came the big moment, after I had plugged the SD card into its slot, changed the TV input to HDMI and flicked the switch on my power supply (which is actually a charger for a Blackberry phone).

A few seconds later after remembering to breathe I saw lights on the board and writing on the screen. After 30 seconds or so it said something like "Performing one-time initialization. Rebooting now" and then restarted itself. 30 seconds later I had the login prompt so I entered my user-name as `pi` and password as `raspberry`. Once I had pressed Enter, a few more lines scrolled up the screen and I had a `pi@raspberrypi:~$` prompt. Enter `startx`, and the screen cleared. A few seconds later I had a desktop that was white with a large raspberry in the middle of it.

Another cup of coffee was made to celebrate, then the serious job began: learning to use this new machine. The get-started guide at *element14.com/raspberrypi* proved useful: it is aimed at non-techies without being patronizing.

Getting onto the internet proved easy. I plugged the network cable in before I started it up the next time. Once it was at the desktop I clicked on the Midori web browser, and when I typed the address of my server into the addressbar it went straight to it. The first problem showed up then: my little animated Gif of my eyes does not render correctly.

Python beckons

One of the first things I will do with my Raspberry Pi is to learn to program in Python until the port of BBC Basic is sorted out.

I have just found out there is a port of the Arduino software based on Python. The original uses Java and does not work on the Pi. This will be of use to me, as I do a lot of experimenting with my Arduino.

I will experiment with the port of RISC OS 5 when I get some time to play.

Noel (Moss) Rolfe *moss@mosschopps.org*

Moss lives in Bridgwater, not far from Archive HQ in Glastonbury. He gets to both the Bristol and Wessex user groups when he can. Useful chap to know if you have pipe or car problems.

Kit needing new homes – Due to ill health Moss is looking to downsize his collection of Acorn BBC and RISC OS computers. He is also downsizing his library. Please email him for details: *stuff@mosschopps.org*

Alpha Risc os on the Pi

Chris Hall

I was most intrigued when I first heard of the Raspberry Pi and, despite the difficulties, managed to get my order in by 08:15 on February 29. Keen to get Risc os working on it, I ordered the AcornC/C++ compiler again – this time it was only £40 from Rool, my earlier copy at £99 from Castle being too old to upgrade.

Leap forward at show

Chris Hall's downloadable alpha version of Risc os for the Raspberry Pi was withdrawn just as Archive was going to press, to make way for a more advanced version to be released at the London show.

By the time my Pi arrived on May 7, there were still difficulties with the USB drivers, so Risc os could not support keyboard, mouse, network or external storage. It had been demonstrated on an alpha board, using a serial keyboard to interact with it, but I decided to wait until the USB had been sorted out before trying it.

First outing

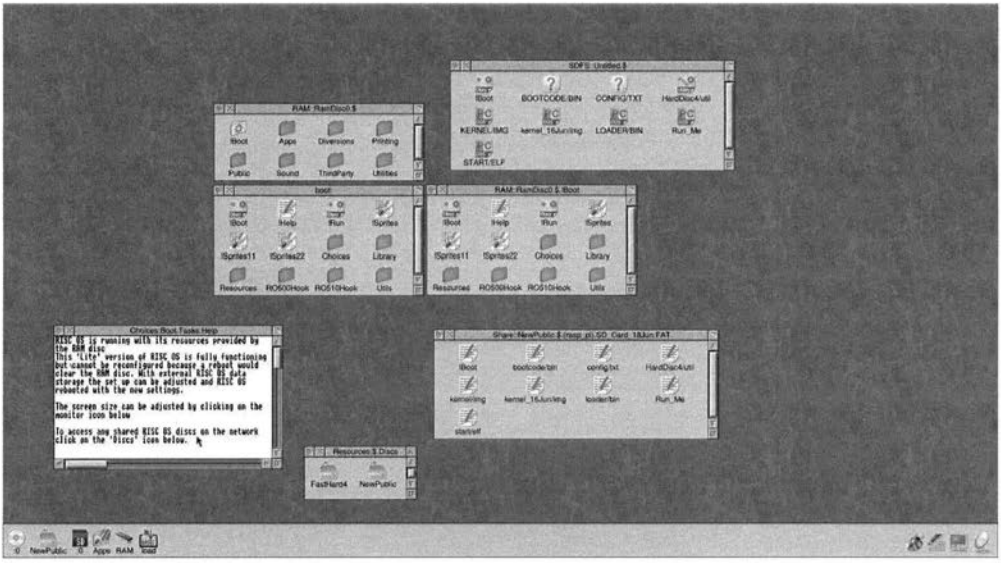
I took it to the Bristol user group on May 9 but just on its own – no point in showing off Linux as I don't know anything about it. By June 5 the first pre-built rom image for the Pi had appeared on the Rool site and it had been announced that SDFS was working on the OMPA3 port. It took me until June 7 to get the right firmware on the SD card and the right config.txt file so that Risc os would start up with the supervisor prompt. USB was now working and Risc os could recognize keyboard, mouse, USB pen drives and the network.

```
RISC OS 127MB
ARM1176JZF-S Processor

No keyboard present - autobooting
Supervisor

*sdfs
*run !boot
**! This file will be run using '*Exec Run_Me'
*
**! either directly or from *Run !Boot
*
**Ex
Dir. SDFS::Untitled.$ Option 02 (Run)
CSD  SDFS:"Unset"
Lib. SDFS:"Unset"
URD  SDFS:"Unset"
KERNEL/IMG      WR/      DOS        04:36:10 16-Jun-2012     6 Mbytes   BOOTCODE/BIN    WR/      DOS          11:34:
CONFIG/TXT      WR/      DOS        00:57:00 14-Jun-2012     79 bytes   HardDisc4/util  WR/      Utility      23:32:
LOADER/BIN      WR/      DOS        11:36:00 07-Jun-2012   266 kbytes   Run_Me          WR/      DOS          06:15:
START/ELF       WR/      DOS        11:36:14 07-Jun-2012 1999 kbytes   !Boot           WR/      Obey         00:00:
*
**basic
ARM BBC BASIC V version 1.51 (C) Acorn 1989
Starting with 113533180 bytes free
>
>REM Allocate 64Mbytes to the RAM filing system
>SYS "OS_ChangeDynamicArea",5,64000000
>QUIT
*
**! Set file type of default HardDisc image
**settype HardDisc4/util Utility
*
**! Copy this self-extracting image to the RAM disc
**copy HardDisc4/util RAM::0.$.* ~c ~v
*
**! Select the RAM filing system
**ram
*
**! Extract the necessary 'vanilla' boot image
**run HardDisc4/util
*
**! Move it into the root directory of the RAM disc
**copy HardDisc4.* * ~c d r ~v
*
**delete HardDisc4
*
**delete HardDisc4/util
*
**!Run Boot
```

The first version (June 16) that could boot up using only the SD card: the file !Boot on the SD card, filetype Obey, runs a script called Run_Me, which copies the Harddisc4 image in a utility file onto the ramdisc, expands it and boots into it (see next picture).



The RISC OS desktop. The directory shows the SD card and all the files needed for the Raspberry Pi to start up.

The default filesystem was 8 (ADFS) rather than 26 (ScsiFS), so it would not boot from a USB pen drive (which appears as Scsi drive 0) and there was no way to tell it the "C-mos" settings to use when starting up. The display was hardwired to 1920×1080 but I was intending to use a monitor this size anyway.

Fast boot

On booting up I therefore got the screen:

```
RISC OS 127MB
Arm1176JZF-S Processor
No keyboard present - autobooting
Supervisor
*
```

What a marvellous effort by those involved! And it boots in only six seconds from power-on!

Not yet usable

Typing `*SCSI` then `*Run !Boot` would boot into the desktop if you had a pendrive formatted as RISC OS with the boot structure on it, but the bottom 5% of the screen including almost all of the iconbar was not visible. So the machine was not really that usable at this stage. One thing I was hoping to do was to get everything needed to boot RISC OS onto the SD card so that it could boot up with no external storage other than the SD card.

All the files needed

By June 15 I had put together a description of the various files that should be put onto a pendrive so that RISC OS could be booted up using the pre-built rom. I copied a utility file from the pendrive to the ram disc using a startup script that would add the necessary files to get a network into action.

Just at that point a pre-built rom for the Pi appeared that had working SDFS.

I therefore compiled my own rom from the sources, setting SDFS as the default filing system and using a recommended bodge to make all of the iconbar display. I put

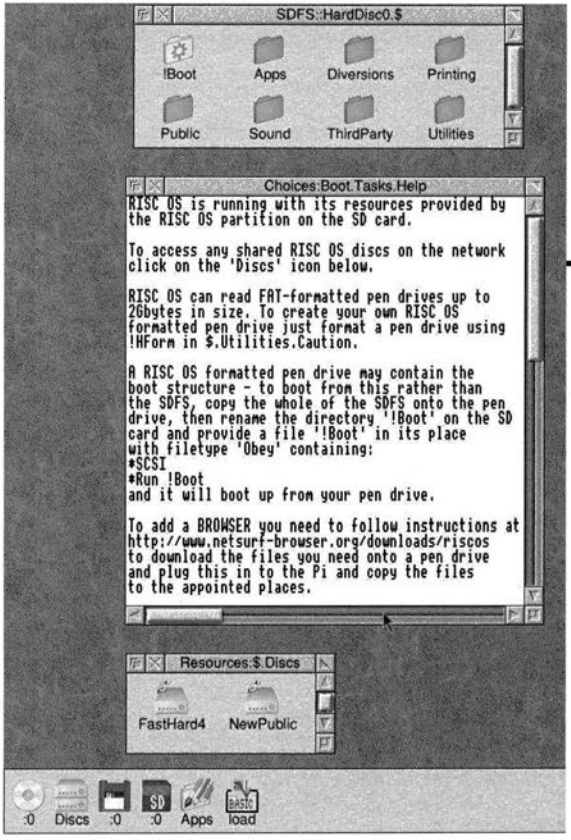
together the boot structure into a compressed utility file so that it could be copied from the SDcard into the RamFS, expanded there and then use the RamFS as the boot drive. Once I had done this, I put together the necessary image file so that an identical SD card could be generated by others and offered it as the first alpha distribution that would support RISC OS on the Pi. This was June 18.

With some help from others, by June 22 this SD card image now contained a RISC OS partition with full boot structure as well as the necessary Fat partition so that the Pi would boot up into the RISC OS desktop with the SD card providing the main storage. This was now reconfigurable as the boot options could be changed and the Pi rebooted.

Smaller and faster

The image was a mere 25 megabytes or so (it unzips to 240M to be written to the SD card using Win32Diskimager) giving RISC OS more than 200M of free storage on the card. RISC OS boots up in 19 seconds from power-on using a Sandisc 2G Extreme III SD card. That's quite impressive compared to Linux, which has a 900M SD card image and takes a couple of minutes to get to the commandline!

Benchmarks updated to include the Pi are at *svrsig.org/Bmk1S.jpg* and at *riscosopen.org/forum/forums/5/topics/466?page=5#posts-12654*


SDFS::HardDisc0.$
!Boot
Apps
Diversions
Printing
Public
Sound
ThirdParty
Utilities
Choices:Boot.Tasks.Help
RISC OS is running with its resources provided by the RISC OS partition on the SD card.
To access any shared RISC OS discs on the network click on the 'Discs' icon below.
RISC OS can read FAT-formatted pen drives up to 2Gbytes in size. To create your own RISC OS formatted pen drive just format a pen drive using !HForm in $.Utilities.Caution.
A RISC OS formatted pen drive may contain the boot structure - to boot from this rather than the SDFS, copy the whole of the SDFS onto the pen drive, then rename the directory '!Boot' on the SD card and provide a file '!Boot' in its place with filetype 'Obey' containing:
*SCSI
*Run !Boot
and it will boot up from your pen drive.
To add a BROWSER you need to follow instructions at http://www.netsurf-browser.org/downloads/riscos to download the files you need onto a pen drive and plug this in to the Pi and copy the files to the appointed places.
Resources:$.Discs
FastHard4
NewPublic
:0
Discs
:0
:0
Apps
!Boot

[Brian Carroll reported: "I am running the Chris Hall alpha RISC OS 5.19 very happily on my Raspberry Pi. It is very fast compared with my StrongArm RiscPC." —Ed.]

Have fun!

Chris Hall *chris@svrsig.org*

Computing has been my main hobby since 1979, but an interest in railways is a close second. I am a chartered mechanical engineer and a volunteer signalman and signalling maintenance technician on the Severn Valley Railway.

◀ *A little more effort and here we are with a much nicer setup on my Raspberry Pi. The RISC OS and the Fat partitions are separated, so that all you can see under RISC OS is the RISC OS stuff. I made this "alpha" distro available for download so that other users could start up in RISC OS on the Pi.*

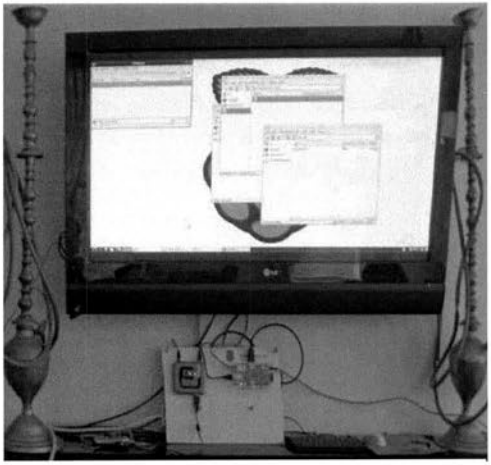
The only problem was finding an SD card that RISC OS can use: it is rather fussy at present. Some suggestions for SD cards that RISC OS will like are on the ROOL site.

Raspberry Pi at Rougol

Bernard Boase

The June meeting of Rougol (the Risc os Group of London) was as well attended as any I can remember: about 25 people, three of whom had brought four Raspberry Pis along to the meeting place, the Blue-Eyed Maid pub near London Bridge. (At least two other members possessed a Pi at that date.)

Both Julian Fry and I brought our Pis. The first photo shows mine after it had played its MP3 files and Julian had booted his alpha Risc os. He ran Artworks to display the well-know Alan Burns picture of an MG Midget. He has tried several apps with various degrees of success; Techwriter, for example, at the time of this report loads then crashes.



Bernard Boase's Pi is mounted on a wooden board that has a four-way power extension fitted behind. The power switch for the Pi is easily operated, thus avoiding frequent handling of the Pi's USB power input socket. Also on the board is a powered USB hub. The Pi is displaying on the pub's wall-mounted TV via HDMI, and is running Debian Linux Music Player to play the Mendelssohn Octet from MP3 files on a USB stick through the TV speakers.

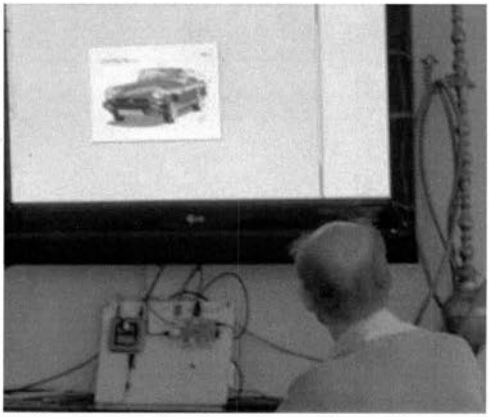
Leo White describing how he got Risc os onto one of his Raspberry Pis.

From Linux to Risc os

Leo White brought both of his Pis: one running his Python robot control under Linux, the other with which to demonstrate the process of "converting" a Linux Pi to a Risc os 5.19 Pi. He ran through the method of obtaining and loading Debian Squeeze Linux (described at *raspberrypi.org/downloads*), then ran the script he had developed to automate conversion to Risc os. The details are in his post on page 9 of the Rool forum topic (*riscosopen.org/forum/forums/9/topics/997*), where he also gives a download URL for the script itself from *mybigideas.co.uk/files/GetRISCOSForRPI.sh*

He said that when Risc os is doing heavy input or output to USB files (such as when recompiling the OS) it has a tendency to lock up. But such problems are to be expected in an alpha version of the system.

Using a recent version of Rool's Norcroft



Julian Fry showing Artworks working on Bernard's Pi (note that no iconbar is showing).

C compiler (compatible with the Arm v6 instruction set) he had recompiled the OS rom image on his Iyonix in order to hack the display parameters so that the iconbar is not obscured (as it still was in Julian's Pi).

(I will be interested to hear what Rob Johnson says about using GCC for Arm v6. The Pandaboard has Arm v7.)

Leo then showed a video clip from Big Buck Bunny (*bigbuckbunny.org*) being played by the RISC OS application Kinoamp, though without sound.

Big graphics, little Arm

People are looking forward to RISC OS having access to more of the Broadcom chip's GPU functionality, and to the port of Aemulor so as to be able to run 26-bit software. Keith Dunlop commented that it's almost more accurate to describe the hardware as being a graphics processor with a 700MHz Arm11 embedded in it!

There was talk also of mass-storage partitioning, sizes of SD cards, and ADFS which I don't yet appreciate, and indeed many details of Leo's demonstration which remain to be understood until I come to try them for myself.

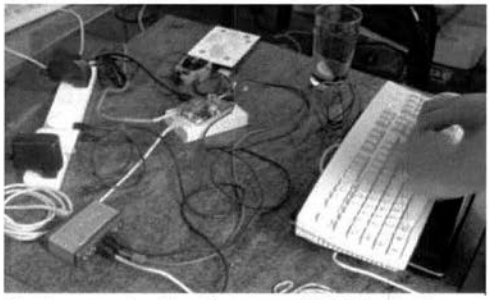
There was a rumour that David Ruck might be updating Armalyser to help ports to Arm v6 (but I'm out of my depth here!).

Jam-crashing

Leo was going to try to attend the London Raspberry Jam session (already a sold-out event) on the following Wednesday on the strength of being able to demonstrate RISC OS (*raspberrylondon.eventbrite.com*).

Judging by that ROOL forum thread, there is considerable activity and the situation changes by the day, with the objective of having RISC OS officially accepted by the RPi Foundation in time for the autumn educational launch. So I'm not sure how Archive can contribute, though a new readership surely beckons.

Bernard Boase *b.boase@bcs.org*



Leo's setup showing Pi, powered USB hub, ethernet connection, USB harddisc and USB keyboard.

Leo's Pi with RISC OS recompiled with fix to make iconbar visible, running Kinoamp, but no audio yet.

The Riscos shows of 2012

A number of show reports remained unprinted in the long gap since Archive 23:4. For readers with Archive in their hands at the London show, this might serve as a guide to what to look out for. —Ed.

Wakefield

Ron Briscoe

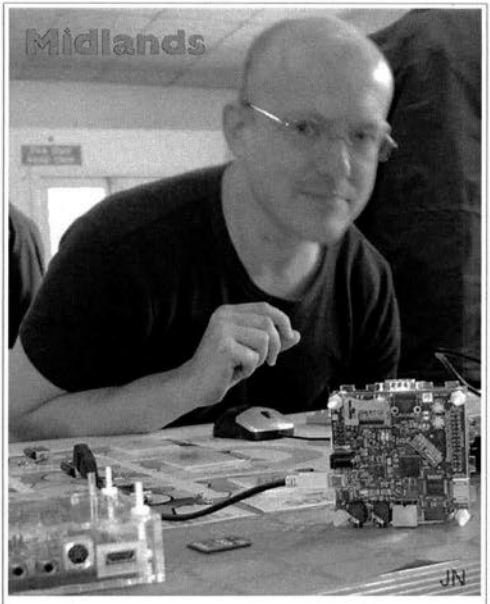
My 2012 trip to Yorkshire on April 28 started out with my apprehensively waiting for the estimable Doug Webb outside the Holy Name Church in Great Barr, near the A34 in Birmingham where I live. I was apprehensive: I have heard rumours that if you stand outside said church too long the clergy drag you in to confess your sins, and I can't spare not that vast amount of time.

Doug turns up and we are soon on our way to the show. We make very good time, as Doug has *not* got a new satnav to try out and thus we don't get lost. We make such good time that we decide to have breakfast in a little cafe just five minutes' walk down the road from the Cedar Court Hotel where the show is being held. This cafe is so close to the hotel entrance that it is farther to walk from the entrance to the upstairs rooms for the show.

On entry

As usual the Wrocc members *[Wakefield Risc os Computer Club]* have made an excellent job of signposting the route to the show and we are soon parting with the necessary spondulicks to gain entry. I try my charms on the lady claiming to be the mother of Chris Hughes – a lady far too young. These charms fail and I have to pay my entrance fee anyway.

Moving swiftly into the main room, Doug and I split up, I to pay my Wrocc membership renewal, a deed that enables me to threaten to turn up to a meeting at Wakefield one day. I also pay for the new **Wakefield Back Catalogue CD**: the complete collection of Wrocc magazines from 1983 to 2012 – a superb read, especially while awaiting the next issue of Archive. *[Ouch. —Ed.]*



Adrian Lees taking a close look at the Beagleboard on the Rool stand at the Mug show. R-Comp announced that Adrian, who first wrote Aemulor 10 years ago for the Iyonix, would soon release a new version of **Aemulor** *for the Raspberry Pi and Beagleboard. That means the new 32-bit-only Arm processors will be able to run classic 26-bit software such as Impression and many games.*

R-Comp said its Armini (which is based on the Beagleboard) will have this Aemulor as standard.

The Midlands User Group show was on July 7, at the same venue in Kenilworth as last year.

Archive DVD ahoy!

I wander over to the Archive stand, manned by both the founding and present editors, Paul Beverley and Jim Nagel, to see if I need to renew my subscription to this august magazine. (Why august? Do you think I am referring to the frequency of Archive issues?) *[Ouch. —Ed.]*

No is the answer, and to boot I can have a copy of the hot-off-the-press **Archive Volume 22 DVD** that I had completely forgotten that I had paid for. Another absolute must for the discerning reader.

Saying goodbye to Paul, who is looking very well, and to Jim, who is still sporting the white caterpillar glued under his nose that we first saw at the Southwest show, I wander off to see what else I can purchase.

Locating Steve

The R-Comp stand is very busy – everybody wants to see the new **tablet** computer, R-Comp's main splash for this show – so I go and ask Steve Fryatt on the availability of Locate 2. *[Locate is a free file-finding utility, which you can download from* stevefryatt.org.uk. *I think it is supplied as standard with RISC OS 4.39, where it is invoked by "Find" on the filer menu. —Ed.]* Because Steve has been busy working on Netsurf, toiling backstage for his local drama society, editing *The Wrocc* (the club's monthly magazine) and co-organizing the show, not to mention researching and writing Archive's PD column, he has had no time to polish up his software collection.

I next pay a visit to the Soft Rock Software stand, a.k.a. Vince Hudd, who still has nothing new to sell and proudly says that he still hasn't finished writing the new manual for Webchange. He does, however, show me some rather good pictures that he is planning to include in his next game.

Martin has to wait

I toy with the idea of joining the usual long queue of people waiting to give Martin Würthner money but decide to wait and instead pay CJE Micros to upgrade Photodesk and CDRomFS. I resist strong urges to buy gadgetry.

The Wakefield club has kindly found room for Mug (the Midlands User Group, my more local club) to have a stand – I sidle past it in case some help is required there.

On up to the ROOL stand, which is minus proud new father Steve Revill. His surprise last-minute stand-in was John Ballance, the father of the Iyonix, whom we haven't seen at a RISC OS show since that machine

The R-Comp tablet

R-Comp's tablet, first seen at Wakefield, is powered by an Arm Cortex A8 chip like the Armini. Although its operating system is Google's Android rather than RISC OS, it easily swaps files with RISC OS machines via USB. The tablet has a 10-inch touchscreen with excellent viewing angles; it automatically switches between portrait and landscape when you turn it in your hands.

It can stream music wirelessly to a hi-fi and send content to a TV. All cables and even headphones are supplied. R-Comp sells it for £259 inc Vat, with useful apps set up to run out of the box.

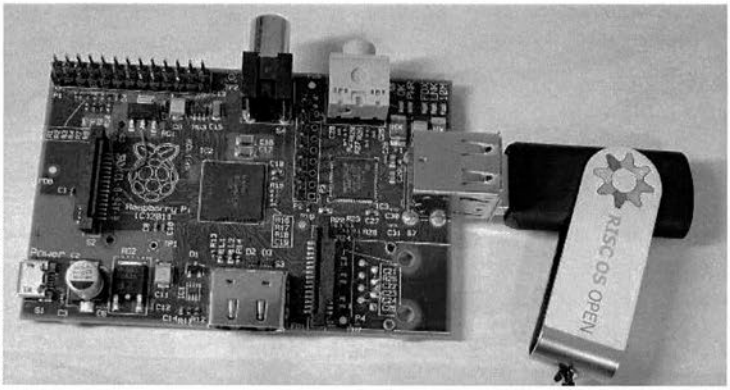


Photo from Riscoscode.com

The Raspberry Pi on the Rool stand was running an alpha version of Risc os. They also had a Pandaboard. Distinctive Risc os-on-USB sticks were on sale.

went out of production. A swarm of locusts has bought up almost all of the Rool merchandise, so I have to pass on empty-handed.

Martin Avison and Nigel Willmott on the Organizer stand do not let me down: they sell me **Organizer on a Stick** complete with a strap thing wherewith you can hang this USB incarnation of Organizer around your neck. All this placed carefully in a brown paper envelope. Slipping the envelope into my bag I feel just like a Metropolitan police officer must feel after lunch with a News International journalist

Organizer on a Stick has the latest version of Organizer, 2.15, loaded on top of the RPCEmu emulator (8.9), which will load quickly and run on all recent versions of Windows. Show price for existing users was £12 and for new users £35.

Tablets for lunch

Suddenly I am surrounded by fellow Mug members: off to the pub for lunch. When we get there the Wakefield member enjoying a well deserved lunch spots me and, bolting his food down, leaves. I will have to carry some indigestion tablets around with me if I have that effect on people. Hearing customers, the bar staff hide but eventually have to give in and serve us. I mentally forgive them for the long wait, as the beer is very well kept.

Fed and watered I resume my quest to find and purchase the latest in Risc os software. I give Martin Würthner cash in exchange for the latest version, new at this show in keeping with his Wakefield tradition. **Artworks 2.X2** has a fine new feature called "artistic lines" – like brush strokes rather than lines of constant width. Lines are drawn in the usual way, straight or curved, and then can

be given variable width, set from a contour dialogue. Colour them as you will, even hatch-patterns or graduated fills, and still edit them as lines.

[Thanks to one of those newfangled tablets, Martin had a further innovation for this show, enabling customers at his stand to pay by card. They simply tapped the tablet to connect to his website via the hotel's wireless and used Paypal there in the familiar way. —Ed.]

Wish I could buy more

I give the charity stand a look-over but as Christine back home is threatening a one-in-one-out policy on Risc os hardware I reluctantly pass over some very interesting-looking stuff and enviously watch other people snap it up. The charity stall made

more than £500 for Wakefield Hospice.

I have to give the Orpheus Internet stand only a fleeting visit because, as good as I have heard Orpheus is, I can't get it on Virgin Cable. Even so, Richard Brown with his usual generosity offers me a sweetie. *[Richard explains that Virgin's network is an amalgamation of all its predecessors' cables, and although BT has opened its network to other operators, Virgin has not. If a would-be customer has only a cable line to the house and not a BT link, Orpheus can provide a new line and place ADSL or fibre on it. —Ed.]*

At the Netsurf stand they are surprisingly pleased to see me. For a small donation I get another sweet and the latest version, **Netsurf 2.9**, on a mini-CD. Life is good.

I stop and talk to Michael Gerbracht, who has come all the way from Berlin, and thank him for the latest version of **Luafox** – the statistical-analysis application written using Lua, the programming language beloved by Gavin Wraith.

Look back and marvel

At the Retro Software stand I marvel at what can be done on with a few megabytes of memory and by today's standards an extremely slow processor. *[Retro Software was able to expand to the neighbouring stand because its intended occupant, Mike Cook with his array of intriguing inventions, had to pull out at the last minute due to a family crisis. —Ed.]*

I then visit the **Wiki Project** stand and spend a few minutes with Trevor Johnson catching up on the latest RISC OS gossip.

Nearly at the show's closing time, I see that the crowd surrounding the R-Comp stand has thinned, so I pop over to see if Andrew Rawnsley and crew have anything new for me to buy. All of his new tablet

Hilary and Matthew Phillips – Sine Nomine – demonstrate the Impact database and Wrangler, which generates and solves numerical puzzles.

computers have been sold by now, and there is no new software but they offer me a very nice deal on a LCD monitor. I am tempted but reluctantly have to say no.

Mug intended to have a meeting after the show to discuss the Midlands show *[which took place on July 7]*. Decisions depended on whether RISC OS would be running on the Raspberry Pi. In the event we didn't take long to discuss the Midlands show and so left to travel home to play with our latest purchases.

Doug surprised me by saying that he had not bought anything from the charity stall this year, owing to decreasing space for keeping stuff. Sounds familiar.

Doug drops me off and I go home – not as you may think to fire up my RISC OS computers, but to listen to Christine's news about her day out at the zoo. A man with designs on a Pandaboard has to play the long game.

Ron Briscoe *ron.briscoe@blueyonder.co.uk*

Although the irrepressible Ron lives in Brum, he's a paid-up member of the Wakefield club as well as of the Midlands User Group (Mug). At home, his charmed wife Christine knows that his RISC OS purchases are for the common good.

Ed adds

Who came farthest to Wakefield? Martin Würthner and Michael Gerbracht from Germany and a couple of French RISC OS users were outnumbered by a Dutch trio: Henri Derksen drove with Charles Deckers and Bob Brand as passengers. "The times of Acorn World with 64 cloggies visiting is alas long past," Bob said.

How many came? Chris Hughes, the main organizer for the 17th Wakefield show, said in the club magazine: "Attendance at the door was up on the past two years, no doubt due in part to news of the Raspberry Pi and Beagleboard developments, and there were certainly a few people saying to me at the end that there had been a nice excited buzz about the place." Stand by for a date to be announced for the 2013 show.

Southwest

John Peachey

The Southwest show 2012, unlike previous years, was organized by Richard Brown of Orpheus Internet and Andrew Rawnsley of R-Comp. They stepped in at the last minute, once Paul Middleton of RISCOS Ltd had decided that he couldn't run the show this year. So a big thank-you to them for rescuing my local show.

The "travelling theatre" was an innovation at this show. Instead of a separate room for presentations, announcements at intervals of a half-hour or so invited everybody to gather round a particular stand for a talk and demonstration. This seemed to work well, and gave all exhibitors a chance to hear the talks too.

The day itself – February 25 – was bright, sunny and very warm, which made it a pleasant trip out to the usual venue just off the M5 in Somerset. The show was quite a bit smaller than previous years. As someone who now gets to only one show a year, that was a disappointment, but it did give me more time to look around the stands. As ever this is my view of the show, and any errors are definitely mine.

Orpheus control panel

First off a quick chat with Richard Brown to find out what was new at Orpheus Internet, and also to find out whether a CPanel-type frontend was going to happen, for customers to control their web domains. *[The off-the-shelf frontend called CPanel is used by many other web providers, including Neil Spellings's Purley Hosting. —Ed.]* Richard said Orpheus has settled on a product called Blueonyx and he was testing the software. Orpheus will likely need to find a new home before the new frontend comes into play, as costs at the current server farm are steadily rising.

Richard also said that there would be a rationalization of prices, making it much simpler for people to choose the right hosting package. This is likely to bring some enhancements for personal subscribers.

Netherlands

The annual day-long get-together of the Big Ben Club, the national Acorn user group in the Netherlands, took place again in May 2012, at the same venue in Koog aan de Zaan, just north of Amsterdam. People come from France, Belgium and Germany to join the Dutch.

Henri Derkson reports that the stars this year were the Belgian owners of revision-C Beagleboards and the member who got RISC OS working on his Pandaboard.

R-Comp's Armini

At my next port of call, R-Comp had its usual array of machines and staff were ready to discuss anything and everything.

The main focus, of course, was **Armini**. When I first saw the machine, at the Southwest show a year earlier, it was very much a prototype and Andrew Rawnsley was unwilling to advertise it as a proper replacement for those still using a RiscPC or Iyonix. But a year is a long time in computing terms, and the machine has had three upgrades of its operating system and the arrival of DVD drive. Andrew is now quite happy to say that it could be a viable alternative for RISC OS users looking to the future.

New for this show was a minute module that properly implements C-mos *[battery-backed memory to retain basic settings when the machine is switched off]* for the Armini's motherboard. The upgrade consists of a card measuring only about 12×5×6mm; it simply plugs onto a header pin on the main board. RISC OS 5.18 is needed as well. This, I was told, is a big step forward as it solves a number of niggling problems.

The upgrade is available from both R-Comp and ROOL to users of all incarnations of the Beagleboard, whether a do-it-yourself build or a ready-made Armini. All new Arminis have the module fitted as standard.

Fireworkz Pro from R-Comp

On the software front, R-Comp had a release of Fireworkz Pro (the integrated wordprocessor-spreadsheet-database) and a new version of Music Man.

A full copy of **Fireworkz Pro** was offered at £35, with an upgrade price of £30 on proof of ownership. I bought a new copy, as I use the spreadsheet side of it a lot.

The new version is much better at importing spreadsheets from Microsoft Excel. It can handle all but the most recent Excel file formats, including workbooks (Microsoft's term for several worksheets in the same file all linked). Workbooks get split up into multiple files in Fireworkz Pro. I asked whether this could be changed so that all the worksheets remained within one file. Andrew wasn't sure that it would be possible, mainly because Fireworkz Pro is not multi-file software, and also because the workbook format does not fit in with the RISC OS way of handling files. However, the existing method of cross-linking between files is being enhanced.

If you buy the full version of Fireworkz Pro you also get a copy of the Windows version for free. Andrew said that when R-Comp took the software over, the PC version had a number of extra features, which have now been worked back into RISC OS.

Music Man 2

Also new from R-Comp is **Music Man 2**. The original was released back in 2005 and had limited ability to "rip" music from a CD and store it in a computer. The new version lets users save music in several formats new to RISC OS, including Ogg and Flac.

On top of this, the software lets users of USB CD drives and Sata drives listen to their music CDs once more, because it can read CDs digitally rather than via the analogue system used by older machines.

CJE's Raspberry Pi kit

CJE had the usual veritable Aladdin's Cave of goodies. Chris Evans was happy to discuss his plans for the soon-to-be-released Raspberry Pi bare-bones computer. CJE hopes to provide a case, power supply and all necessary cabling,

either as a kit or prebuilt, along with other related items that would let you to connect the Pi to almost anything – it could act as a central-heating controller, for instance. Chris showed some dinky 3½-inch screens that can be easily connected to the Pi in its role as a controller. Another item in the kit will be a realtime clock, which will be fixed to the header pins.

At the time of the Southwest show, RISC OS had not yet been fully ported to the device, though progress was being made. Chris said the Pi's graphics and sound capability reportedly surpass the Beagleboard, although this had yet to be confirmed. I think it likely, as Pi is designed to be a computer whereas Beagle is more of a development environment for the mobile-phone market. It will be interesting to see how the two products match up once both have had time to mature.

Wikipedia and RISC OS

I had a long and interesting chat with Trevor Johnson, who is looking for volunteers to help with the **Wiki Project** for RISC OS, especially as 2012 marks the 25th anniversary of the Acorn Archimedes computer, where the Arm chip started. The project is looking for people to locate and cite sources, write clearly and work with others to produce a factually accurate picture of RISC OS and its software.

You don't need to spend a lot of time, but any help would be appreciated. If you have a quiet couple of hours you could go to the wikiproject pages, pick a subject and do a little bit of research (*en.wikipedia.org/wiki/Wikipedia:WikiProject_RISC_OS*).

Archive

On the Archive stand alongside Jim Nagel, our Ed. N°2, was Chris Johnson, who had travelled all the way from Edinburgh. He does the sterling work of converting Archive articles into HTML for the end-of-volume discs. He demonstrated new versions of some of his own software and the utilities he maintains from David Pilling's stable.

Artworks and 'Writer

Martin Würthner was of course at the Southwest show showing the current releases of Artworks and Techwriter.

Artworks 2.XI, which was released at Wakefield 2011 as the "XI" hints, is the first version to be recompiled for Arm v7, and so runs on Beagleboard and Armini. XI also introduced new features such as a style panel, which helps you to apply consistent design across different object types – something I usually get wrong).

Techwriter and **Easiwriter** version 9 was launched at the London show last November. This version automatically updates cross-references and has improved import of Open Office documents, retaining text formats and styles.

Drag'n'Drop

Paul Stewart was on hand with the latest version of his PDF magazine ***Drag'n'Drop***, with various offers and bundles available. Please support this venture, for the RISC OS world is now down to just two magazines. I know that an electronic magazine is not to everyone's taste, but it should be simple to print your own paper copy from the PDF.

Roundup

• Steve Revill and the ROOL crew happily talked about new features of **RISC OS 5** and the Beagleboard. They had various versions of RISC OS and the **RPCEmu** emulator on a USB pendrive for £10 a pop. It was nice to see ROOL at the show, especially as RISC OS Ltd was not able to attend.

• Vince Hudd of Soft Rock Software had a stand advertising various useful links, and also had the latest versions of **Webchange** and **Quicksand**. It was easy to see from the various proposed release dates for the software that he is a programmer of true genius: the release dates were described as "possibly in the future". I wish I'd had a pen with me to note few of them down, for use on my own deadlines!

• Archive's own Steve Fryatt was on hand – his first time at a Southwest show – demonstrating the latest versions of **Cashbook** and **PrintPDF**. The latter produces PDFs from any RISC OS application that can print – a great help in publishing your work in the wider world.

• John Norris was showing off not only his **bellringing** software and tower but also a tasty line in jams and preserves. The software was being demonstrated on a BBC Micro when I went by, but there was also a RiscPC version and one for Windows.

• For David Snell, from Exeter – with **Webwonder**, **Procad** and **Funfysics** – the Southwest is his local show.

Here's to 2013

In a newsgroup after the show, Brian Bailey "thought it was a very good show. It was notable for the spirit of openness and cooperation that RISC OS has needed for so long." Martin Würthner, who came all the way to Somerset from Germany, seconded the thought: "I did enjoy it and found it worthwhile attending."

All in all it was a typical Southwest show: quiet, but a good chance to have a natter to the few who still work to keep RISC OS alive. Again a big thank-you to Andrew Rawnsley and Richard Brown for rescuing the show. I hope that it was a big enough success to warrant a show next year.

John Peachey *john@jpeachey.co.uk*

John lives in darkest Gloucester. His interest in computing started when he played with a BBC B in the sixth form. His leading claim to fame was winning the Wimp Basic programming back in 1991. He says it has been downhill ever since!

[Andrew Rawnsley of R-Comp told Archive just before we went to press that a Southwest show 2013 is indeed on the cards. Stand by for a date to be announced. —Ed.]

hint&tip Finding a lost file in an over-full folder

This idea has just occurred to me and it has saved me spending ages hunting for a file in a directory that had loads of files in it. I knew the filename was something to do with "keystroke", but it didn't start with "k".

Here's what I did:

• Open a textfile in StrongEd.

• Over the directory, click menu and then Select all.

• Shift-drag the contents of the directory into the new textfile.

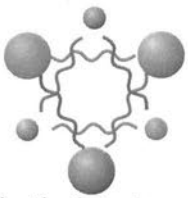
• Now use StrongEd's interactive find to locate the file.

I got it in seconds!

The Shift-drag has the effect of loading the (leaf and) filenames of all the files and directories into the textfile.

Paul Beverley *paul@archivepub.co.uk*

Charm – a practical guide



At the London show Pete Nowosad will demonstrate a new Mandelbrot image explorer *(below)*. This application, based on the toolbox, is included in the latest release of Charm, version 2.6.2.

Peter Nowosad

This and follow-on articles are intended to promote interest in and understanding of the Charm language for RISC OS computers and emulators. As the author of the language, this is a goal I am keen to foster, particularly among the younger generation of Raspberry Pi owners who are looking to learn a little about the mysterious world of programming.

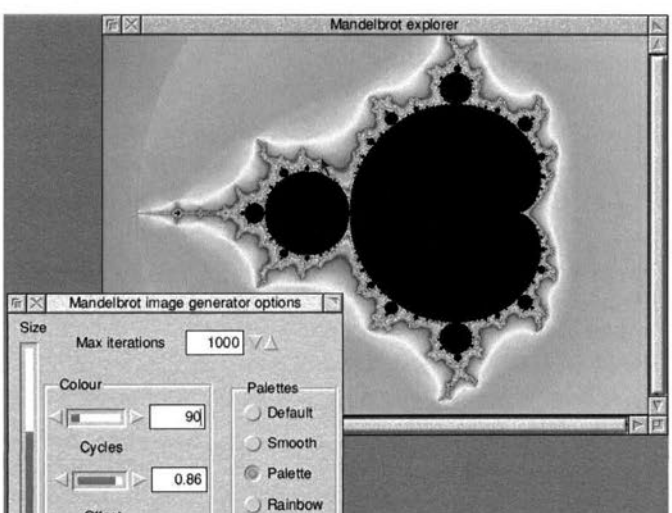
The Charm tools are light on resource yet powerful and highly suited to an agile development environment. Applications of any complexity can be rapidly developed in small incremental changes without waiting around for code to build; for instance the whole of the Charm distribution can be rebuilt in well under a minute!

As of May last year, more than 1,000 unique users from around the world have visited the Charm website, and the majority made return visits. Unsurprisingly most interest is from Britain, the Netherlands and Germany, where RISC OS is still actively used. More exotic locations include Bolivia, Kazakhstan and Mongolia.

Gavin Wraith wrote an excellent introductory review of Charm in *Archive 23:4* (thanks, Gavin!). Following on from this and encompassing the recent extension of the language into object-oriented territory, I would like to cover the practical aspects of Charm – in essence how to install and use Charm, augmented by discussion and examples of the basic syntax and semantics of the language. I hope this will encourage people to take up writing Charm programs for themselves.

Installation

Charm is freely distributed under the Gnu Public Licence as a simple zip file from the Charm website *charm.qu-bit.co.uk*, where you can read more, get in touch with me through the RISC OS Charm Language users page on Facebook and view screenshots of the various Charm demos.



The Mandelbrot explorer, programmed in the Charm language. Another example is on the cover of this issue; you can download the file.

To install Charm, simply unzip it and copy it to where you want to keep it on your file system. (Should you need a de-archiving utility, David Pilling's !Sparkplug is also available for download.)

If you don't have a RISC OS machine available to run Charm, you can freely install the RPCEmu emulator under Windows or Linux and run RISC OS and Charm in a window (or full-screen if your graphics card is up to it). I would suggest using the RISC OS 5 release from RISC OS Open Ltd for the Iyonix as a suitable base operating system.

Do remember to keep an eye out for new releases as the language and its libraries and demos evolve.

Tools

The set of Charm tools for RISC OS contains the following principal applications:

• **!Charm** – A desktop shell application for running the tool set. It supports drag-and-drop for files and folders and logging of commands, reporting errors and enabling vector floating-point (VFP) support for targeting Arm cores that support it.

• **edit** – A general-purpose editor that is useful for writing and developing Charm source code.

• **armc** – A compiler that transforms a source code file written in the Charm programming language into an object file in binary form.

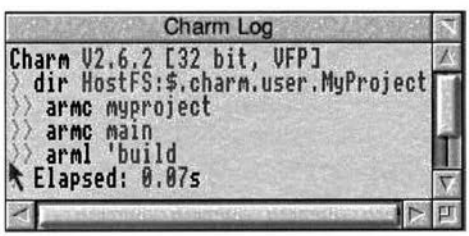
• **arma** – An assembler that transforms a source code file written in Arm assembly language into an object file in binary form.

• **arml** – A linker that combines Charm object files into an executable RISC OS application or module.

Since the compiler allows direct in-lining of Arm assembler code, the arma application does not get much use these days, but is included for completeness.

New projects

Creating new template projects in Charm is now easy using the !Newproject utility application. Simply select New Project from the menu, name the project as you wish (default is `Myproject`) and drag the folder icon to the folder in which you wish it to reside. You can then build the project by dragging the project folder onto the Charm icon. You should then see the following in the Charm log:



All the required project folders are created automatically, namely:

• **src** – Charm language source code files

• **arm** – Arm assembler source files

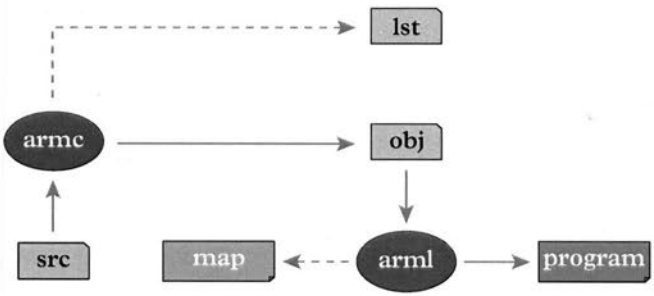
• **obj** – object files created by the compiler or assembler

• **lst** – "list" files created by the compiler or the assembler (if requested)

The key folders are the src folder where the Charm source code files are stored and the obj folder where the corresponding compiled binaries are stored. The arm, lst directory and map files are typically of use only for debugging purposes (I may cover how to do this in a later article).

Applications that are to live inside an application folder are usually linked directly to the correct !Runimage location

from inside the project 'build file using the `program` command.



Charm's key folders. Dotted lines come into play for debugging.

Modules

The concept of modules is key to an understanding of Charm. (A word of caution: *Charm modules are not the same as RISC OS modules.*) Each module is introduced with the keyword `module`, and each source file that Charm compiles must contain one and only one module definition. It may, however, import exported declarations from any number (including zero) of other modules on which it is dependent.

Typically these references are to other modules in the containing project, on occasion modules from other projects and library modules that provide key functions such as managing windows, the keyboard and screen. The latter are further documented on the Charm website. The order in which these modules are compiled is determined by the project 'build file which also specifies the name and location of the linked application or module.

Get started

So now for the first few snippets of the Charm language. Each project must also contain one module in which a ~start procedure with one of two specific signatures is exported.

If no command-line parameters are required, the ~start module is like this:

```
module Main
{
    export proc ~start ()
    {
    }
}
```

If command-line parameters *are* required, the ~start module is like this:

```
module Main
{
    export proc ~start (int argc, ref
array ref array char argv)
    {
    }
}
```

In this second form, the number of command-line arguments and a nil-terminated array of their string values is passed into the start procedure.

A practical project

If you are up for completing a short Charm practical, I suggest replacing the default !NewProject MyProject module with the following code to output the first dozen factorial numbers:

```
import lib.Out;

module MyProject
{
    proc factorial (int n) int
    {
        if n <= 1 return 1;
        return n * factorial (n - 1);
    }

    export proc start (ref array ref
array char argv)
```

As usual in Archive's narrow columns, tightly packed lines ndicate a single line of code.

```
    {
        for int i := 1 step inc (i) while
i <= 12
            Out.vdu.num_fld (i, 2).str
("! = ").num (factorial (i)).nl ();
    }
}
```

This code illustrates the use of recursion to calculate each value from the previous value via multiplication starting with the fact that 1! = 1.

Finally, as an exercise, try changing the program so that the first 20 factorials can be calculated without running into the 32-bit limitation on integer size. (Hint: return a real from factorial and use `.float` instead of `.num` to output it.)

Next time

The next article will cover the Charm type system, name scopes and basic programming constructs related to selection, iteration and calling of procedures.

Peter Nowosad *pete_nowosad@msn.com*

Perhaps I'm obsessed with producing Mandelbrot images because it takes me back to my youth in the 70s: I used to own an equally psychedelic Paisley shirt and wear a cravat.

Iyonix power supply: watch it

Jim Nagel

Before reflashing a new version of RISC OS onto your Iyonix, be sure that its power supply is up to spec. Otherwise you may find you have erased rather than reprogrammed the flash memory.

The warning comes from John Ballance, the chief engineer in the Iyonix project at Castle Technology. "Most Iyonixes were bought 10 years ago now. In the PC world you'd be very unlikely to be running a machine, unless it was much cherished, more than a few years old.

"With the best will in the world, electrical bits do dry out – especially the capacitors. We now see more power supplies fail than we did two years ago, because they're that much older. A new power supply costs £30 or £40; it's not worth paying a serviceman an hour's labour to fix an old one."

The 5-volt red-to-black rail should show a decent 5.06 on a voltmeter, he said, "give or take a tiny bit". The orange-to-black should read 3.3V without much variation. "If not, the power supply is not doing what it should do: replace it."

No frontend software test is possible. A machine that seems to be working fine is no guarantee that its power supply is still in spec, he said. The power supply is likely to be well out of spec before you see any evidence in using the computer. (Several users have reported that their first symptom of a power-supply fault was that their Iyonix began refusing to connect properly to their network.) John said he found one Iyonix nearly 50 percent over voltage. He shut it down quick, surprised the motherboard survived.

All is not lost if you do unluckily erase the rom. "Salvation is to contact me *[see p.59]*, and I do a reprogram service. We get two or three a year typically, but I do maintain the capability to deal with it." The charge will involve carriage and probably a new power supply, and being without your Iyonix for several days.

A different viewpoint

Gerald Fitton

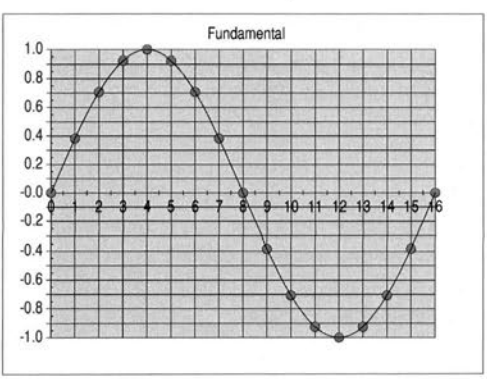
Using different points of view is a writing skill. Marks are awarded for the use of viewpoint.
—The National Curriculum

There are two senses in which the phrase "Change one's view" can be used. One is the sort of change of view experienced by St Paul on the Road to Damascus where he changed from hating to loving.

My article this issue is not about the sort of change of opinion that moves a slider along the same scale (a scale running from Hate to Love or a scale with graduations representing Worst, Worse, Bad, Good, Better and Best) but it is better described by the phrase "A change of viewpoint". A couple of examples of what I mean will help.

Files: see p.2

I remember the first time I saw a photo of the Earth taken from outer space. Until then the only pictures I had seen of the Earth were artists' impressions. Without exception, all these artists' impressions depicted the Earth without any clouds. Similarly I was greatly surprised when I first saw satellite photographs of the area where I have lived for 50 years.



In this article I shall discuss (with worked Pipedream spreadsheets) such a radical change of viewpoint. In my example I shall start with the viewpoint that Time is my independent variable; then I shall change my viewpoint by introducing a different independent variable.

A sound wave

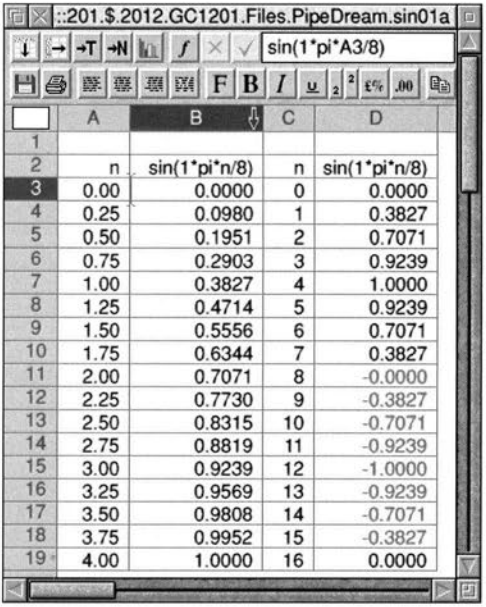
The graph called Fundamental has, as its independent variable, time; the dependent variable is the instantaneous air pressure at a point in space. This graph shows one cycle of of a sound wave for which the time axis is marked 0 to 16 instead of from 0 to 1. Why? Be patient. All will be explained.

Although the time units could be anything at all, let's think of them as milliseconds. A complete cycle of this sound wave takes place over a periodic time of 16 milliseconds. Although this graph shows only one cycle of this sound wave, please imagine that it carries on repeating this same 16-millisecond cycle well into the future – as it did in the past.

The formula I used for this Fundamental sine wave is

$$p = \sin(2\pi t/16) = \sin(\pi t/8).$$

I drew the sine wave using the Pipedream spreadsheet shown in the screenshot at right.


::201.$.2012.GC1201.Files.PipeDream.sin01a

sin(1*pi*A3/8)

	A	B	C	D
1				
2	n	sin(1*pi*n/8)	n	sin(1*pi*n/8)
3	0.00	0.0000	0	0.0000
4	0.25	0.0980	1	0.3827
5	0.50	0.1951	2	0.7071
6	0.75	0.2903	3	0.9239
7	1.00	0.3827	4	1.0000
8	1.25	0.4714	5	0.9239
9	1.50	0.5556	6	0.7071
10	1.75	0.6344	7	0.3827
11	2.00	0.7071	8	-0.0000
12	2.25	0.7730	9	-0.3827
13	2.50	0.8315	10	-0.7071
14	2.75	0.8819	11	-0.9239
15	3.00	0.9239	12	-1.0000
16	3.25	0.9569	13	-0.9239
17	3.50	0.9808	14	-0.7071
18	3.75	0.9952	15	-0.3827
19	4.00	1.0000	16	0.0000

The smooth curve is drawn using the first two columns (which have a time interval of 0.25ms) whereas the red points are obtained by plotting columns C and D.

The physicist or sound engineer may look at this periodic waveform of pressure against time and deduce many interesting things from it. However, we mere human beings do not experience this phenomenon as pressure variations. Our perception of a sound wave is not one that is comparable in any way with the way we sense time-dependent pressure variations in our hands and arms when using a hammer drill. The way in which we perceive the pressure variations of a sound wave having a 16-millisecond period is as a pure sinusoidal tone with a frequency of 62.5 Hz.

We have changed our viewpoint from one of *time* dependence (the air pressure is time-dependent) to one in which we regard the *frequency* of the sound wave as all-important.

GCSEs and A-levels

When we plot graphs at GCSE-level they are all about variables. We might plot a graph of y against x for $y = ax^2 + bx + c$ having been given numerical values for a, b and c. This graph is a parabola. The variables are x and y; a, b and c are called *parameters*.

Also at GCSE-level we might plot a graph of $y = A\sin(\omega t + \psi)$ for different values of A, ω and ψ. This graph is a sine wave similar to the one in the graph called Fundamental.

When we plot graphs at A-level we no longer concentrate on variables such as x, y or t. We change our viewpoint; we concentrate on the effect of changing the parameters a, b, c for the parabola and A, ω, ψ for the sine wave.

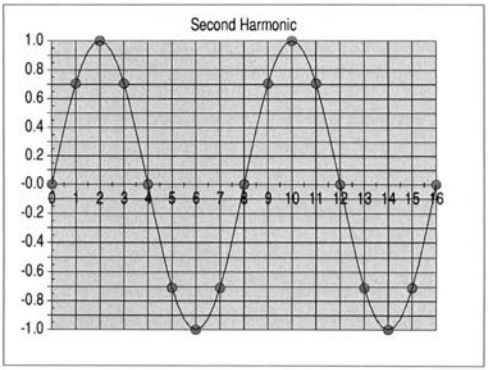
Sine-wave parameters

The three parameters of a sine wave, A, ω and ψ, change the shape of the graph. Let's consider how these changes in the parameters affect the shape of our Fundamental graph.

The effect of changing the parameter A is relatively easy to describe. As A increases from 1.0 the maximum value increases. We experience this subjectively as an increase in the volume of the musical tone. We call A the amplitude of the sine wave.

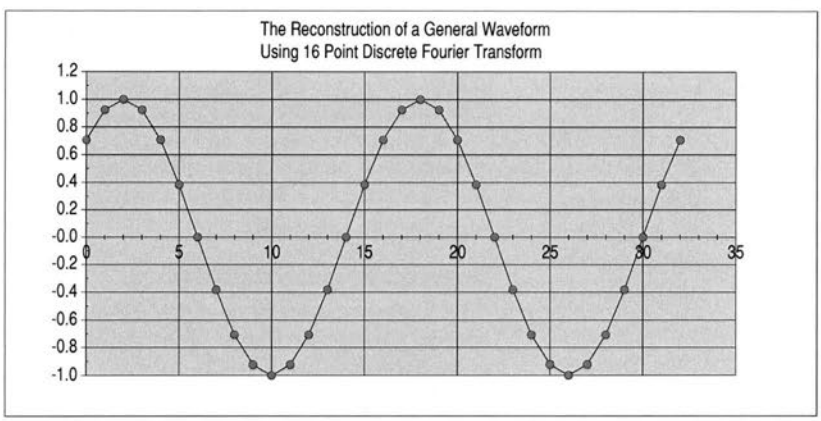
The parameter ω is called the frequency. The pitch of the tone we hear rises as ω is increased and falls as ω is reduced. Doubling the value of ω doubles the frequency of the note we hear from, say, 62.5 Hz to 125 Hz.

Have a look at the graph labelled Second Harmonic *(right)*. You will see that over 16 milliseconds the pressure wave in the air has completed two cycles rather than the single cycle of the graph labelled Fundamental. The formula for this Second Harmonic graph is $p = \sin(2\pi t/8)$ rather than the $p = \sin(\pi t/8)$ of the Fundamental. We have doubled the value of ω.



The parameter ψ is perhaps the hardest to explain.

In the graph below, called Reconstruction of a General Waveform, I included two cycles of a sine wave which, as before, has amplitude 1.0 and a 16ms period.

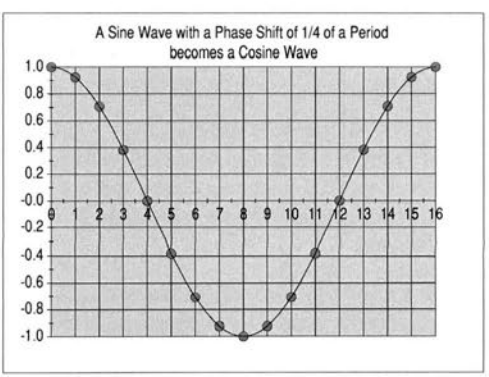


However, when the time (along the horizontal axis) is $t = 0$, the pressure (the vertical axis of the graph) is not zero as it was before but it is about $p = 0.7$. This shift of the graph to the left ($p = 0$ when $t = -2$) is called the *phase shift* (or phase offset) of the sine wave.

A single ear can do nothing with such a phase shift but, if we receive two copies of the sine wave having different phase shifts, one sound wave in each ear, then we can use the perceived phase difference to determine the apparent direction from which the sound originates. Our aural mechanism, which includes the use of our brain, is very sensitive to the phase difference that we receive between our two ears. Isn't our brain clever?

If we phase-shift our sine wave through 1/4 of a 16-millisecond period, our sine wave turns into the corresponding cosine wave! The formula for this wave is

$$p = \sin(\pi(t+4)/8) = \cos(\pi t/8).$$



An algorithm for our ears

Our ears (or rather the whole aural system including the way our brains process sound waves) receive a *time*-dependent *pressure* waveform and then, using a very clever algorithm, we, our ears and brains, convert this time-dependent viewpoint to one in which we experience only the *frequency* or frequencies of the sounds we hear.

There is a mathematical algorithm for changing our viewpoint from time-dependence to the viewpoint for which frequency rather than time is the independent variable. From a mathematical viewpoint we are trying to extract m, the frequency, from the formula $p = \sin(\pi mt/8)$. The name given to this extraction of the frequency from the pressure waveform is called "finding the Fourier Transform".

I shall try to explain, with a somewhat simplified example, the way this algorithm is used to extract a set of frequencies, the values of m (in our case the Natural Numbers 0 to 8), from a general periodic pressure waveform

$$p = f(t)$$

for which p is a function of the time, t.

The maths bit

The maths to extract the frequencies from the time-dependent waveform requires some basic familiarity with the integration of trigonometrical functions. In our case we need to integrate the information in the pressure waveform over a complete cycle in order to find the frequency.

For those of you who want to get into this Maths Bit, the extraction depends upon the fact that the integral of

$$\sin(\pi nt/8) \times \sin(\pi mt/8)$$

over a whole period (exactly a whole period) is nearly always zero. It becomes something other than zero only when $n = m$.

It is important to note that m and n are not Real Numbers (from the set $\mathbb{R}$) but that they have to be Natural Numbers ("counting numbers" from the set $\mathbb{N}$). Only when m and n are Natural Numbers such as 0, 1, 2, 3, 4, 5, 6, 7, 8 does the function $p = \sin(\pi mt/8)$ have the same period of 16 milliseconds as the Fundamental. For this mathematical trick to work, the period has to be 16 milliseconds, just like the Fundamental; consequently m and n have to be Natural Numbers.

I show the "hard" (integral calculus) maths bit as an Equasor drawfile *(below)*. You can see that we start with the pressure waveform, $p = \sin(\pi mt/8)$, and, by using this algorithm, we are able to extract the frequency we want by finding the n for which this integral returns the value 8. When we have chosen the right n then the m we want is given by $m = n$.

$$\text{For } m, n = -8, \ldots 0, \ldots +7$$

$$F(n) = \int_{t=-8}^{t=+8} \sin(2\pi nt/16) \times \sin(2\pi mt/16)\,dt$$

$$= 0 \qquad \text{for all } n \neq m$$

$$F(n) = 8 \qquad \text{when } n = m$$

If you are into the integration of trigonometrical functions, you will know that the way to integrate $\sin^2(t)$ is by using the identity $\sin^2(t) \equiv (1 - \cos(2t))/2$. When $\sin^2(t)$ is integrated between the limits 0 and $2p$ it returns the value p.

Rather than consider a general waveform at the moment, I shall concentrate on the waveform given by

$$p = \sin(4\pi t/16) = \sin(2\pi t/8).$$

This is the waveform I have called the Second Harmonic. The graph of this function appears earlier in this article. The frequency we wish to extract is $m = 2$, the 2 cycles per 16 milliseconds which appears in the formula $p = \sin(2\pi t/8)$.

The Fourier Transform method of extraction for our 16-point waveform is as follows:

Create a set of sine waveforms having the format $q = \sin(\pi nt/8)$ using values of n from $n = 0$ to $n = 8$. The nine waveforms, q, are nine different functions because of the nine different values of n whereas (in this example – a more complex example later) the waveform p is only a single function having one single value of m. In our example the waveform from we wish to extract the frequency, m, is $p = \sin(2\pi t/8)$; the value of $m = 2$.

Using each of the nine waveforms for q in turn, multiply $q \times p$ and integrate the function $q \times p$ over a single 16-millisecond cycle of the pressure waveform. Nearly all of these integrals will return zero but when the particular q is chosen for which $n = 2$, namely $q = \sin(2\pi t/8)$, the integral of $q \times p$ returns not zero but, wonder of wonders, it returns the number 8.

We have found the Fourier Transform of $p = \sin(2\pi t/8)$. This Fourier Transform has the simple result that the frequency, m, is 2 cycles per 16 milliseconds.

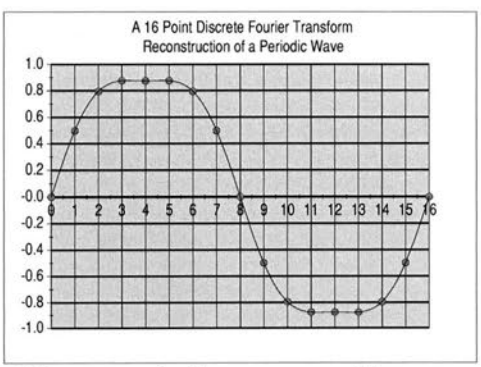
More than one tone

Now let us consider our pressure waveform to be not a single pure tone but a mixture of two pure tones. For this example I have chosen a pressure waveform having the formula

$$p = \sin(\pi t/8) + (1/8)\sin(3\pi t/8).$$

This pressure waveform is the sum of two sine waves. The second term is the Third Harmonic of the Fundamental, and to us this third harmonic will sound like a note two octaves above the Fundamental with an amplitude that is one-eighth of the Fundamental waveform's amplitude.

The second term of the pressure waveform formula, $p = \sin(\pi t/8) + (1/8)\sin(3\pi t/8)$, has an amplitude that is an eighth of the amplitude of the Fundamental. As you will see from the graph with the rather long title "A 16-point Discrete Fourier Transform reconstruction of a periodic wave", this Third Harmonic flattens the peaks of the waveform.



Different musical instruments add different amounts of either odd or even harmonics; it's these extra harmonics that give each musical instrument its unique, distinguishable voice.

If we combine two tones that are the second and third harmonics without the Fundamental, then the chord produced is called "a perfect fifth". Chords consist of many frequencies often played by different instruments. Using our ears and brain we

are able to separate the different frequencies and to distinguish one instrument from another.

[Philip Perry's "Elements of music?", a series that ran from Archive 17:2 to 18:9, explores these ideas in copiously illustrated detail. An overview of his series is in Archive 18:1. *—Ed.]*

Finding the Fourier Transform

What we have to do is to "find the Fourier Transform" of

$$p = \sin(\pi t/8) + (1/8)\sin(3\pi t/8).$$

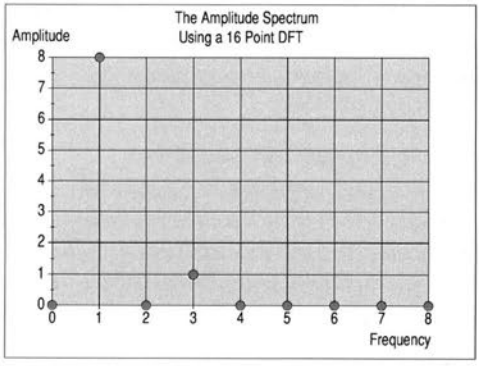
The Fourier Transform is the spectrum of frequencies contained in the original waveform.

In order to do this and provide a general solution I have constructed a set of spreadsheets which you can download from the Archive website among the files for this issue. The chart entitled "The amplitude spectrum" shows the amplitude of each of the nine frequencies.

In this chart the independent variable is the frequency; the range is from 0 to 8 cycles per 16 milliseconds. The dependent variable is the amplitude of each frequency present in the composite waveform.

The Fundamental has an amplitude of 8 at the frequency $m = 1$ cycle per 16 milliseconds. The third harmonic, $m = 3$, has an amplitude of 1.

By finding the Fourier Transform of the pressure waveform we



have successfully changed our viewpoint from that of a time-dependent pressure wave to regarding the sound wave as a spectrum, a mixture of frequencies. This process is what our ears and brain do with a composite air-pressure waveform; they find the Fourier Transform of the sound wave and we hear the mixture of tones. We do not experience the sound as being a time-dependent waveform but as music or speech having different frequencies and their tonal harmonics.

Finding the Fourier Transform of a

ipeDream: hostfs::201.$.2012.GC1201.Files.PipeDream.dft-12.general16-01b *

sin(1*pi*(A3)/8)+sin(3*pi*(A3)/8)/8

	A	B	E	F	G	H	I	J
1		Original		Reconstructed			Spectrum	Spectrum
2	t	Waveform	t	Waveform		m	Sin	Cos
3	0	0.0000	0	-0.0000		0	0.0000	-0.0000
4	1	0.4982	1	0.4982		1	8.0000	-0.0000
5	2	0.7955	2	0.7955		2	0.0000	-0.0000
6	3	0.8760	3	0.8760		3	1.0000	-0.0000
7	4	0.8750	4	0.8750		4	0.0000	-0.0000
8	5	0.8760	5	0.8760		5	0.0000	-0.0000
9	6	0.7955	6	0.7955		6	0.0000	-0.0000
10	7	0.4982	7	0.4982		7	0.0000	-0.0000
11	8	-0.0000	8	-0.0000		8	0.0000	-0.0000
12	9	-0.4982	9	-0.4982				
13	10	-0.7955	10	-0.7955				
14	11	-0.8760	11	-0.8760				
15	12	-0.8750	12	-0.8750				
16	13	-0.8760	13	-0.8760				
17	14	-0.7955	14	-0.7955				
18	15	-0.4982	15	-0.4982				
19	16	0.0000	16	0.0000				

waveform is the mathematical equivalent of using a physical piece of kit called a spectrum analyser. Our ears act as a spectrum analyser.

The 'general' spreadsheet

A screenshot of the main spreadsheet of the linked set of the three spreadsheets that I used to find this frequency spectrum is on the previous page, "General16-01b". Most of this spreadsheet has protected cells having a yellow-brown colour (is this colour called ochre?). You cannot change the content of those cells without removing the protection. However, you can enter 16 values, any values you choose, into the block B3B18. Then, in columns I and J, the frequency spectrum of a periodic waveform fitting those 16 points will be returned. In column F I reconstructed the original waveform from the spectral components. (More about this in another article.)

The screenshot here contains the formula

```
sin(1*pi*A3/8)+sin(3*pi*A3/8)/8
```

in cell B3; it is replicated down to B18. The spreadsheet as shown in this screenshot calculates the values of p (the pressure) corresponding to the formula I used for the previous section, namely $p = \sin(\pi t/8) + (1/8)\sin(3\pi t/8)$. Columns I and J contain the spectrum of this wave.

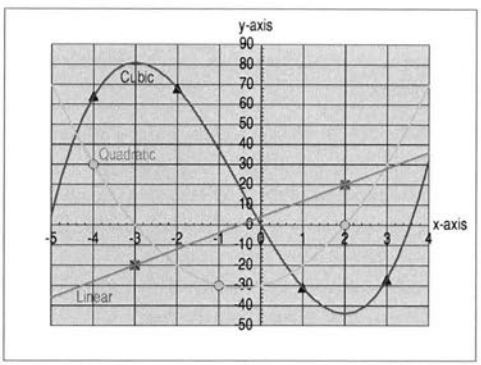
If you download a copy of this (and the two supporting spreadsheets) and load all three into Pipedream (or Fireworkz) then you will be able to enter your own 16 values into B3B18 and find both the sine and cosine components of the spectrum of your 16-point periodic function. Both sine and cosine components are needed for reconstruction of the amplitude and phase. (More about amplitude and phase in another article.)

Curve-fitting

In an earlier article, "Curves are better" (*Archive 22:8*), I explained how matrix methods could be used to fit a polynomial of any order (any power) to a set of points. I used as an example the cubic polynomial shown in the graph called "y-axis". It has the formula

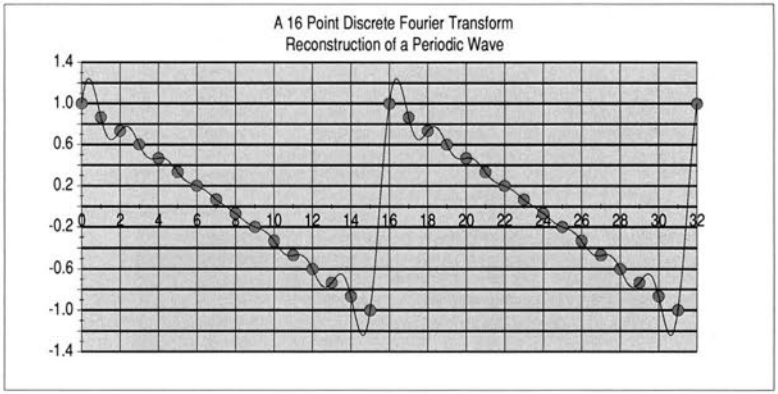
$$y = 2x^3 + 3x^2 - 36x + 0$$

and it passes through the four points (−4, 64), (−2, 68), (1, −31) and (3, −27).



The most serious disadvantage of fitting a polynomial to a set of points is that nearly all polynomials shoot off to infinity once you extrapolate to values of x that are outside the range of given points. Most of the ordinary variables we come across in this world don't do that! Polynomials are OK for interpolation but they are not appropriate for extrapolation.

I was inspired to write this series of articles about Discrete Fourier Transforms after reading the excellent article by Nicholas Cutler in *Archive 23:2* called "The secret life of algorithms". Nicholas' article referred to an algorithm for calculating Fast Fourier Transforms. The FFT to which he refers is a fast practical method of calculating the Discrete Fourier Transform (the sine and cosine spectral analysis) of a periodic wave.



Fourier Analysis lets us create periodic functions that repeat endlessly even when the values of the independent variable are extended outside the range of given points.

It is usual for the number of points (I chose 16) to be an integer power of 2 (such as $2^4 = 16$, $2^{10} = 1024$, $2^{14} = 4096$) because it makes the calculations easier. The calculation of a Discrete Fourier Transform is easier because it allows the use of the algorithm known as the Fast Fourier Transform which was described by Nicholas.

I chose 16 points because 16 is small enough to allow the waveform to be analysed and then synthesized easily while being large enough to demonstrate all the principles involved in finding a Discrete Fourier Transform. Using 16 points allows us to find a spectrum but with frequencies only up to the 7th harmonic. A much larger number of points, for example 1024 points, would allow us to find the spectrum up to the 511th harmonic.

If you look at the graph above called "Reconstruction of a periodic wave" you will see the original 16 red points that I used to create two cycles of the spectrum of frequencies. Also you will see a smooth black line. This is the line showing the "in-between" values returned by the analytical extension of the periodic curve using the periodic functions from the spectrum we calculated. This reconstruction does pass through all 16 points. However, it doesn't exactly match the linear (straight-line) Ramp Wave of the original 16 points but deviates on either side of that straight line – but it deviates only between the 16 points.

In a future article I shall show you how a 16-point Discrete Fourier Transform recreates an almost square wave that is not as square as the original.

The limitation of using 16 rather than more points is that the Spectrum of frequencies we can use to recreate the waveform is limited to the 7th harmonic. This limitation can be compared with the limitation of using an 7th-power polynomial to find a curve passing through 8 points. The power of the polynomial is one less than the number of points. In the case of the Discrete Fourier Transform the highest harmonic frequency included in the reconstructed waveform is one less than half the number of points: $(16/2 - 1) = 7$.

The reason why the black line deviates from the straight line passing through the 16 points is that we have selected only 16 points. If we forced our black line, the reconstruction of the original waveform, to conform more closely to the straight line then we would need more points and a larger number of harmonics.

You will see that I included a 17th point at $t = 16$ which is repeated at $t = 32$. This is to show you that the reconstructed waveform is periodic and, unlike the best straight line through the original 16 points, it does not shoot off to infinity outside the range of times for which we have data.

Another way of drawing smooth curves through points is by using Bézier curves. Just as in the case of polynomials and, in this article, Discrete Fourier Transforms, the actual process of calculating the parameters of Bézier curves requires the use of matrix methods. Matrix methods, which are built into both Pipedream and Fireworkz, do make the formulae used to compute these parameters in a spreadsheet simpler to write and simpler to execute.

Gerald Fitton *archive@abacusline.co.uk*

A very long time ago, in the 1970s, one of Gerald's inventions, a heat pipe used as a cooking skewer, appeared on *Tomorrow's World* and later on *Pebble Mill at One*. These two 40-year-old cooking skewers still work well.

Search Gerald's website for blue-eyed blondes

Gerald has added a Search facility to his website, which contains PDFs of all the articles he has written for Archive: *downloads.abacusline.me.uk/pdfs*. He elaborates:

"If you type `Spamstamp` into the search box then you will find yourself presented with a list of all the articles which I have written for Archive which contain the word 'Spamstamp'. The search engine finds seven PDFs containing Spamstamp; I have not reread them so that, by now, they may be total rubbish!

"Of course, if you vaguely remember that I once wrote an article about 'Blue-eyed blondes' then this search engine will find you the four articles in which I use this example of conditional probability.

"Also, I find that I used the phrase 'Philosophical Bit' in 10 articles and Epistemology in 14."

All of Gerald's articles are on the Archive CDs (and DVD) too, of course – as plain text, as cross-referenced HTML and as the Impression files that printed the magazines.

small ads

Magazines: *The Micro User* (6 binders containing one year's issues each), *Acorn User* (2 binders with one year's issues each), *Archive* (unbound, many complete years), many other unbound issues of these, hundreds of cover disks. **Hardware:** Acorn A5000, BBC Master, Acorn RiscPC, Master rom, manuals for many eras of machines. **Software:** wordprocessing, DTP, publishing, accounting, many others.

All this amounts to four heavy boxloads, so it would need to be collected from me in Bath. Free if someone takes the lot; otherwise I will consider offers for individual bits.

Simon Hall *simonhall@eurosoba.com*

I want to sell my **Iyonix Panther**, which I never seem to use since I bought VirtualRPC. The Iyonix has a 120G disc and 512M of ram. With it I will bundle **Impression Publisher** (with disks and manual) and **Photodesk** (with CD), licensed copies of **Aemulor** and **HID**, which allows reprogramming of the keyboard. It occasionally hesitates at startup, needing the reset button. This was a common quirk of the Iyonix caused by a problem with initializing the graphics card. When running, it is entirely stable. CJE seems to be asking for £540 or so for a similar package. I was thinking of £400 plus carriage, or collect from Brighton.

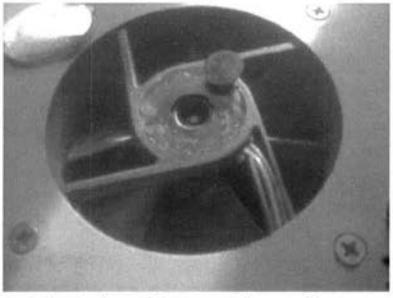
Arthur Quinn *arthur@bellacat.com*

Hint&tip Noisy RiscPC fan

Does your computer sound like a jumbo jet starting up? If so, the fan in your power supply might need a simple service.

You'll need to dissemble your computer, being careful about static. Remove the power supply and lay it on a towel or newspaper on a table.

See the sticker on the middle of the fan? Very carefully peel it back. Under it you may find a small



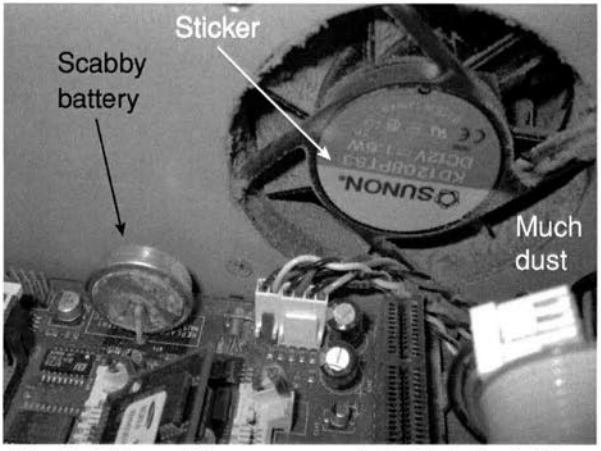
rubber plug. Remove it, and you will see a small shaft with a circlip (snap ring) holding the shaft in place.

Put one drop of 3-in-1 Oil on the shaft. (Do not use WD-40. Bryn Evans says the RS Oil Pen is even better than 3-in-1.)

Now replace the plug if there was one and replace the sticker. The sticker is very important to keep the dust out – oil and dust together work as a grinding paste!

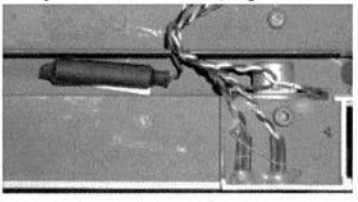
Reassemble the computer and enjoy a quieter machine.

Noel (Moss) Rolfe stuff@mosschopps.org



Ed adds: This would be a moment to deal with the inevitable dust collection around the fan. I just use the nozzle or brush attachment on the vacuum cleaner, all around the power-supply and the motherboard. (Photo from Webmonster.nz)

Whenever you have the machine apart, have a good look at the battery. If it shows the slightest sign of furring or corrosion, replace it pronto – if it leaks it can ruin the motherboard. Another telltale sign of a failing C-mos battery is your desktop clock sometimes showing the wrong time, but by then it mght be too late. Any RiscPC is likely to be 15 years old, so if the battery has never been replaced, even if it seems to be OK, replace it now before problems rear up.



The £3.20 kit from APDL is good value: it's a rechargeable AA cell with flying leads that you solder to the clipped-off legs of the old battery. The new cell then stows away from the motherboard so that if it ever leaks in future it causes no harm. This solution is especially advised for an A3000, whose original cell lies dangerously flat on the motherboard.

Reading problems Sound 32

Jim Lesurf

In my previous Sound article *(Archive 23:4)* I explained that some computer systems (and software) may do a better job of ripping tracks from audio CDs than others. In this article I look in detail at the effect on the resulting files. For the sake of example I have chosen to use a specific set of audio tracks that I recorded a few years ago onto a (TDK brand) CD-R, using my Iyonix and !CDVDBurn.

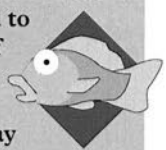
I used CD Paranoia to rip Wave files from this CD-R using two different machines. One was a Shuttle running Linux, using a DVD drive; the other was my wife's Iyonix with its CD drive. On the Iyonix, Paranoia ripped with no reports of any errors or problem. But on the Shuttle, Paranoia listed a string of "+++++" warnings during the second half of the ripping process. The "+" characters are Paranoia reporting "unreported loss of streaming [or] other error in read". So that's a sign that the drive was having problems reading the disc.

In both cases I ran CD Paranoia with the

Battling the greed of corporate dinosaurs

CDDA Paranoia is minimalist open-source software to extract ("rip") compact-disc digital audio directly as a digital file, without going through an analogue stage like some other methods. It is designed to be able to compensate for poor hardware to produce a flawless file. The CD Paranoia logo is called the "Laser-Playback-Head-of-Omniscience".

The Paranoia project is part of the Xiph Foundation, "dedicated to protecting the foundations of internet multimedia from domination by self-serving corporate interests" (see essay at *xiph.org*). The Xiph community produces compression-decompression (codec) routines with quirky names like Ogg Vorbis and Ogg Theora and Flac, which compare favourably with patented rivals such as MP3, AAC and MPeg-4.



Alex Macfarlane Smith did a command-line version of CD Paranoia for Risc os in 2003: *www.archifishal.co.uk.*

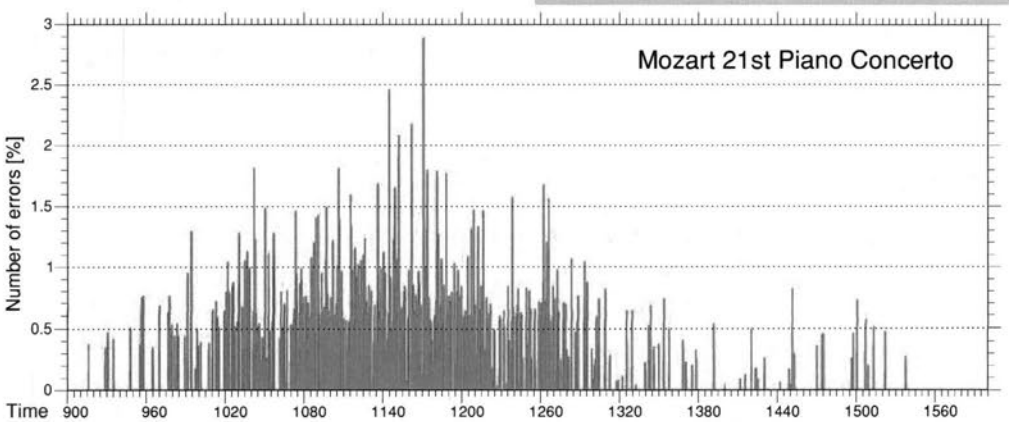


Figure 1 – How many errors (differences) occur during each chunk of 0.1 sec

"-z" option. This tells the program "never accept any less than perfect data reconstruction". Hence it would re-try reading frames of data over and over again until satisfied it has correctly read the disc. So the appearance of "+" characters during ripping doesn't necessarily mean the result has errors. Indeed, you might expect that using "-z" means there can't be any errors if the rip gets to the end of the specified track(s) because Paranoia would have gone on trying until it succeeded – or you interrupted its attempts because it was taking too long.

Same but not the same

The two resulting files were the same size, so contained the same number of samples. That was promising. But when I used my !Wavdifference application to subtract one set of samples from the other I found the resulting difference Wave file wasn't just a series of zero-values. The files were different, so at least one of them must contain some erroneous sample values. This was despite Paranoia showing its ":)" smiley throughout as if happy that it was successful in overcoming the "+" problems on the way.

To investigate further I wrote a new application in the !Wav series that I have called !Wav_compare. This goes though two Wave files and checks, sample by sample, to see if they agree. It then reports what it found. I could then plot the outcome.

The tracks I used are actually a copy to CD-R of a performance of Mozart's 21st Piano Concerto. The analysis analyses the two rips in 100-millisecond chunks. It counts how many samples in each chunk are not the same in the two rips. Figure 1 plots the result as the percentage of such errors in each 100-millisecond period. Note that the horizontal axis has units of seconds measured from the start of the music.

The graph displays only the part of the music from 900 seconds onward because, before that, the two rips are exactly the same. The highest percentage value is just under 3%. For a (stereo) audio CD there will be 2 × 4410 = 8820 samples in each 100-millisecond period. So an error rate of 3% would correspond to around 260 samples being different between the two rips during that chunk.

Figure 2 plots the amount by which the values differ. To simplify things, the plot

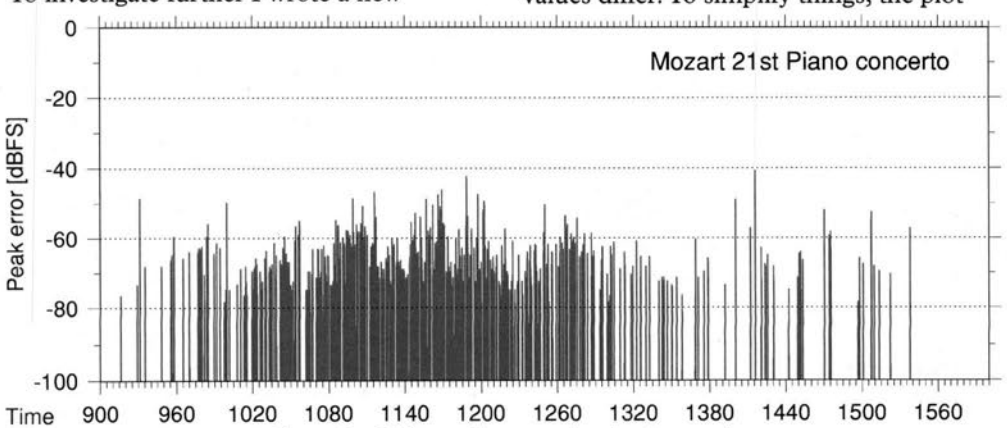


Figure 2 – Difference between values in the two rips

actually shows the biggest difference between rip values it could find during each 100-millisecond chunk. The difference in values is then referenced to the maximum signal level. In effect this shows the discrepancy during each chunk as if it were a brief noise spike of the size indicated. For values around −60dBFS or less it seems unlikely that the effect will be noticed. However, some of the errors reach around −40dBFS, which may be audible.

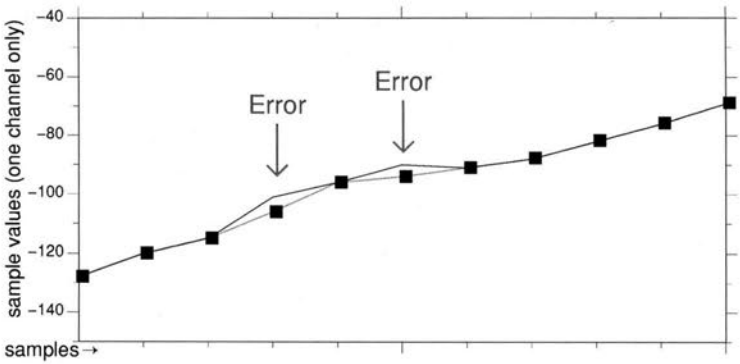


Figure 3 – Detailed view of a couple of error events

Figure 3 shows what individual errors (discrepancies between the two ripped files) tend to look like when examined in detail. Here I have plotted just a few successive samples for a channel from the two files. The two lines plotted are for the two different rips of the same audio channel. I have also put squares on the plot to mark the actual sample values for one of the rips. If the two rips were identical you would see only one line on the graph. But where the rips differ you see two lines. Two errors are apparent in this section. The sample values actually differ by only 3 or 4, which is quite small compared to the 16-bit range of audio-CD values.

On the basis of the above it is tempting to conclude that the errors generally don't really matter much. Typically, the discrepancies affect only around 1% or fewer samples, and the actual change in level is small. So the two versions should sound close to indistinguishable. However, there are some grounds for caution.

Here I've looked at only one example of a CD that gives different results when ripped with two different systems. So it may well be the case that other discs and ripping setups give worse problems. My impression, though, is that if CD Paranoia continues to give its "`:)`" smiley and rips the specified tracks despite using the "`-z`" setting, then the errors may be minor. More serious errors would probably cause Paranoia to stall and go on re-reading the same frames until told to quit. So it wouldn't complete the process of ripping. I've not (yet) checked, but I suspect that when the program gives "`-`" warnings but continues apparently happily, then again the errors may be minor. But all that said, my analysis shows that using the "`-z`" setting and having the rip complete with a "`:)`" throughout does not guarantee the result is totally free of any sample value errors. So you can't be certain that a rip is perfect.

Why only small errors?

There's a curio here. What is the nature of the errors, and how do they cause such – modestly small – discrepancies in values? If the errors were random changes of an individual bit in a raw binary sample data stream then the changes produced would

be just as likely to be large than small. This is because a significant bit is just as big a target as a less-significant one.

Yet the errors I found all tend to produce small changes. This is strange, but it may be a result of the coding systems used for audio CD. The raw "channel bit" data stream on the CD isn't simply the stream of sample values as LPCM. It is encoded and wrapped into a number of layers of error-protection.

So maybe one of the features of that system is that there is a tendency for errors to be small once the encoding, etc, are unscrambled. On the other hand, maybe the changed values are actually being produced by CD Paranoia by a process of guesswork or – more politely – interpolation. That is plausible, but it isn't obvious from figure 3 that this is what is happening.

Which is right?

In the end we are left with a more basic question. How can be we certain which of the ripped versions is "wrong"? The only evidence we have is that the results differ for some sample values, and that one ripping process issued some warning "+" indicators. Taken at face value, the "+" warnings tell us the ripping process was struggling. But CD Paranoia seemed happy that it had read the results OK despite this. So maybe it had, and the rip using the Iyonix is the version that actually contains errors – despite its giving no warnings. Maybe Paranoia on the Iyonix misses problems, and so doesn't correct them, whereas the Linux system did.

To set against that disturbing thought, I can note that for some other audio CD-Rs, the Iyonix system does report either "+" or "–" warnings. So the problem is not that the Iyonix can't actually detect errors. This is encouraging and makes me feel that the Iyonix is getting better results – leaving the Linux setup as the one that is producing some erroneous results. Of course, sometimes *both* might be wrong!

With these questions in mind, I'll finish with another question you may like to ponder. How can we check which system may be in error, and check to see if, at times, both make errors without giving any warnings?

Jim Lesurf *jcgl@audiomisc.co.uk*

During the last few weeks I've been trying to write programs to analyse and convert "HDCD", and copying terabytes of data from one HD to another. At the same time I've been investigating the damage the Olympics has done to the sound and vision quality of the BBC stations that were shoved about to make way for all the people running around fields in their underwear.

Alan Leighton

small ads

For sale in Manchester: **Net100 network card** (NIC), four-port **backplane**, **Hydra** backplane with processors and memory modules, but *no controller card*. **RiscPC** 700 with Strongarm (233), **Viewfinder** (Rage), **Unipod** (full), keyboard and old-style mouse, two slices (knackered), **CD writer**. Offers?

Christopher Self *cself@btinternet.com*

Free to share: PD column

Steve Fryatt

This issue's column breaks with tradition: after my nearly nine years of being at the keyboard it's the first one that I haven't written on an Iyonix. With the demands of the Wakefield show out of the way for another year, I've finally found time to start looking into the use of RISC OS on a Beagleboard. The result is that, for one issue at least, all the software mentioned here will have been tried first-hand using RISC OS 5.18 on an Arm v7 system.

Faster browsing

Apparently not people to break tradition, the Netsurf developers had a new release of their open-source web-browser available for visitors to take away from the Wakefield show in April. This year's **Netsurf 2.9** was being hailed as the "fastest Netsurf ever" – a claim the team could back up with an impressive set of benchmarks and statistics.

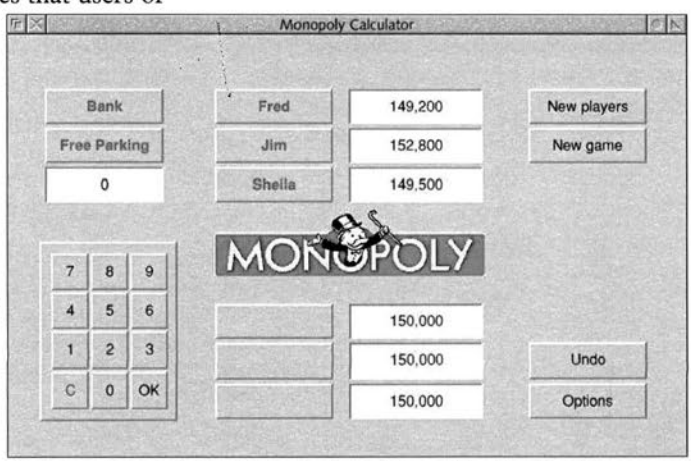
Alongside speed, this stable release formalizes a range of features that users of the test-builds have been familiar with for a while: improved page rendering and handling of internet standards, better image-caching and restored support for "favicons" (those little site-specific logos that appear next to the address in the toolbar). For RISC OS, 2.9 has better support for keyboards and scrollwheels, fixes to drawfile export and the ability to work with external hotlist applications such as Nick Roberts' Bookmaker.

The Netsurf team were selling CDs at the show to help cover costs such as the bandwidth used by their website. The browser is released under the GPL; download either the latest stable release or the latest test-build from *netsurf-browser.org*

A run on the bank

Playing Monopoly can be fun – as long as you enjoy it, of course. The game gained popularity in the 1930s Depression, and many hours then and since have been spent buying up property, collecting rents and passing around brightly coloured banknotes. The last point is a bit of a problem in this modern age of direct debits and bank transfers, though: it's just so *old-fashioned* – not to mention that, if my memory is correct, the bank usually runs out of money just as you manage to build a hotel on Mayfair.

This has clearly bothered Paul Sprangers as well, and his solution – **Monopolycal** –



Paperless Monopoly banking, thanks to Monopolycal by Paul Sprangers.

looks set to revolutionize the way the game is played in RISC OS households. Providing electronic banking for wannabe property tycoons everywhere, it allows for all the movements of money required by the game's rules and tracks each player's balance on screen.

Even if you're one of those who *don't* enjoy Monopoly, Paul's program could still have something to offer. As he noted in his announcement: "Until recently, I never wanted to play Monopoly. Since I created this calculator my children refuse to play. This is what we call a win-win situation."

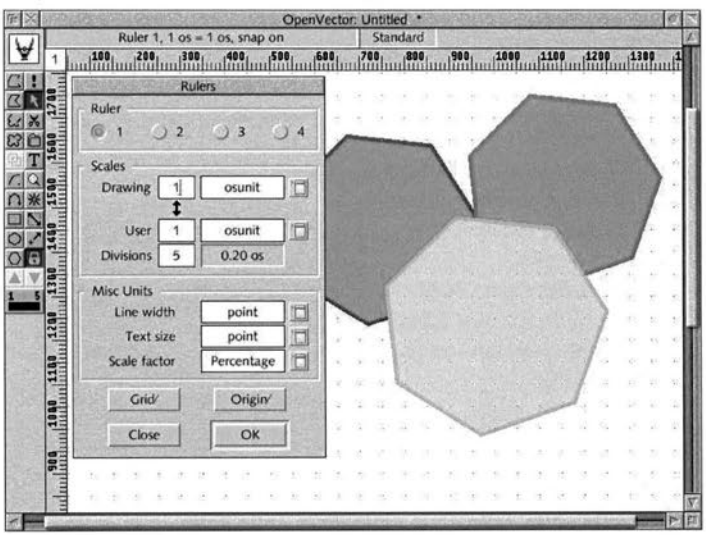
Monopolycal is simple, but does what it sets out to do (although a keyboard option would improve the data entry considerably). If you would like to try it out, it's freeware, from *riscos.sprie.nl*

Graphic improvement

Ever since Chris Martin in Australia started updating Jonathan Marten's **Draw Plus**, **Open Vector** and **Open Grid Pro**, the three vector-graphics packages have been given a new lease of life with a steady stream of improvements over the past couple of years. Draw Plus was, as its name suggests, an "upgrade" to Acorn's standard !Draw and it was initially released as freeware. Quirky and powerful, it evolved into Vector: an advanced graphics package sold in the past by 4Mation to a different target audience from Artworks. Grid Pro was a sister package with facilities for creating many different kinds of grid.

Chris's most recent set of updates bring the three applications to version 3.43. Along with fixing a few problems encountered on Arm v7 systems, and a more serious bug in text-to-path that affected all platforms, Chris has improved the user interface in the areas of path-editing, object-dragging and libraries. Resizing objects with their control points has been made easier when grid lock is on, as the points now jump to the nearest grid lines and so provide visual feedback. A new shortcut sets the origin when editing a path, and an Alt-click forces any objects under the pointer to be ignored.

The library interface has been improved in a number of areas (the release notes have all the details), and the handling of layers has been rationalized. There's also a new measurement unit, `osunit`, which corresponds to a RISC OS screen unit and can be useful for creating pictures intended for on-screen use.



Drawings can now be created in Draw Plus, Open Vector and OpenGrid Pro using OS units for measurement.

In a column this size, it's hard to do justice to the many improvements Chris has made to the three applications: they would require an article all to themselves. If you're new to Draw Plus, Open Vector and Open Grid Pro, or if you last used them before they became open-source, the best thing to do is to download them along with the changelog from Chris's website and have a look. All three pieces of software are released under the GPL and can be found at *users.on.net/~belles/software/openvector*

Users of Open Vector may also be interested to know that there's now a mailinglist, courtesy of Richard Torrens. Subscribe by sending a blank email to *openvector+subscribe@torrens.org.uk* – Chris Martin reads the list, so it's a good way to give feedback and ask for help. Richard has also produced a guide to Open Vector's hotkeys: *torrens.org.uk/RO/Vector/Keys.html*

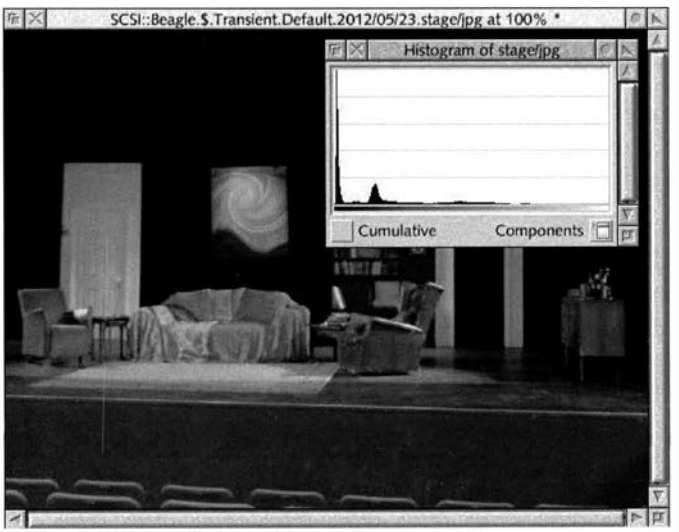
Bitmap and vector viewer

For viewing images, there are a number of free options. However, in recent years David Thomas' **Private Eye** has seen regular updates. The release of 2.99 earlier this year finally made the improvements since 2008 available to those unwilling to run "test" versions of the software.

At its heart, the software is an editor and viewer for bitmaps and vector graphics. It can cope with sprites, JPegs, Gifs, PNGs, Artworks and drawfiles. Files whose RISC OS types are not recognized will be inspected when dragged to the iconbar and re-typed if necessary; other formats will be passed to Computer Concepts' FFG system (TransFSI, TransTiff++ and so on) if available. For each of the formats it understands by itself, Private Eye can be configured to claim files on doubleclicks or leave them for their respective native editors when available. Other options are available for each format and depend on the capabilities of the format and viewer.

Once an image has been loaded, there are a range of options for scrolling or panning and zooming in or out. Two image-info dialogues make details of the image (both the original and the converted sprite, when applicable) clearly available. For JPegs it's easy to get access to the Exif metadata with information about the camera and the photograph; for graphics converted into sprites, histograms of brightness, red, green, blue and alpha components can be displayed.

Private Eye is more than just a viewer. For bitmaps it also offers a range of processing options. When displayed directly, JPegs can be rotated through multiples of 90°



Private Eye is a well-featured image viewer and editor.

without loss of quality; when converted to sprites (which can be done automatically), and for all the other bitmap formats, more features are found in the effects dialogue.

A range of effect tools is provided, which can be sequenced in the dialogue and applied to an image in realtime. Each effect is blended with the current image using a level control that ranges from −100% to +200%. The extended range allows for some interesting effects: clearing to black with a level of −100% allows an image to be brightened, for example. The manual explains this, but the immediate nature of the tools (plus the ability to undo an effect by simply deleting it from the list) also allows for easy experimentation.

Morekeys offers a neat way to enter characters that aren't on the keyboard.

The range of effects includes clearing to a colour (which can be used to tint an image), greyscale, blur and sharpen. Private Eye has options for expanding the dynamic range of an image, equalize its histogram and adjust various curves (gamma, brightness, contrast, midpoint, bias and gain).

Though it's not competing with the dedicated bitmap-editors, Private Eye is a very competent image-viewer with some useful and comprehensive effects thrown in. The current version is happy on modern Arm v7 systems, and is now released under a BSD licence (the sourcecode is available from *riscos.info*). The application itself, plus a well-written PDF manual, can be found at *davespace.co.uk/risc.os/privateeye.html*

Alternative keypresses

Entering non-standard characters at the keyboard can be difficult. Several different schemes have been proposed over the years, from the in-built Alt combinations provided by RISC OS (such as Alt-S to type a German eszett, **ß**) to many third-party modules that often made sense only to their authors. Those with no memory for arcane keypresses have the option of !Chars (or Martin Würthner's Xchars), but these are somewhat cumbersome for use when typing.

Back in March, Rick Murray posted to the newsgroups about a new utility called **Morekeys**. Based on an idea from John Williams, it occupies a middle-ground between arcane key combinations and selecting from lists of characters. To type an accented character, hold down Ctrl and Alt then press the unaccented key (so to type **ï** or **í**, press Ctrl-Alt-i). A small bar of possible options appears at the base of the screen; select one with the cursor keys and "type" it by pressing Return.

Besides accented characters, a few keypresses offer additional symbols: Ctrl-Alt-M gives a range of maths symbols, for example, and Ctrl-Alt-Q leads to a selection of sexed quotation marks. Mappings are supplied for Latin 1 and Latin 2, along with UTF8 for those experimenting with Unicode on RISC OS 5.

In practice, Morekeys is a utility that is much easier to use than to describe. In the past I've always stuck with Xchars (or Ovation Pro's built-in equivalent) because I don't tend to remember Alt-key combinations beyond those that I use regularly. Rick's choice of hotkeys is so intuitive that I can imagine myself giving his utility a go for a while. Although only in alpha-state at present, it has been stable enough on my system. Find it at *heyrick.co.uk/software/morekeys*

Beagleboard utilities

One of the things that owners of Beagleboards will have to do from time to time is update the version of RISC OS installed on their machine. Unlike other RISC OS hardware, these diminutive systems store the "rom" image on a removable SD or MicroSD card. However, the SDcreate tool supplied by ROOL assumes that the card to be updated is *not* the one currently being used to boot the system.

In many cases, this isn't actually a serious problem: it can be a good idea to keep several cards with known setups on them, particularly when following the latest developments of the current "unstable" OS. Swapping cards is just a case of switching off the machine and locating the slot (if you know what you're doing, it can sometimes be OK to remove the card while the system is running). Even an Iyonix needs to be rebooted for a newly flashed OS update to take effect.

Being able to update the boot card directly can be useful, however – especially for users without a "C-mos widget", because changing the configuration currently means updating a file on the card. For anyone who needs to, Dave Higton has produced a small utility called **BBflash** that works in a similar way to SDcreate but operates directly on the installed SD card. It may still be necessary to use SDcreate to produce the required files.

Another of Dave's utilities for Beagleboard is **SDactive**, which can check either the boot card or one plugged into any USB port to see if its partitions are "active" and set them if not. This is important, as only active partitions can be used to boot a system: if the files are stored on an "inactive" card, the system will not boot. Both BBflash and SDactive are free to download from *davehigton.me.uk*

Finally, for this issue, another recent update for Arm v7 update came from yours truly when it was reported that Thomas Leonard's **Iconclipbrd** utility – used to implement the global clipboard in text fields on RISC OS 5 – crashed with alignment exceptions turned on. A quick check of the source revealed a few problematic unaligned memory accesses, which have been corrected in version 0.16. Released into the public domain, copies of the new version – which has been used while writing this article – can be found at *www.stevefryatt.org.uk/software*

Steve Fryatt *steve@stevefryatt.org.uk*

Steve not only researches and writes the PD column for Archive, he also edits the monthly newsletter of the Wakefield club, writes software and helps maintain the RISC OS frontend of Netsurf. And toils backstage at the Leeds Arts Centre, currently producing Peter Quilter's *Duets*.

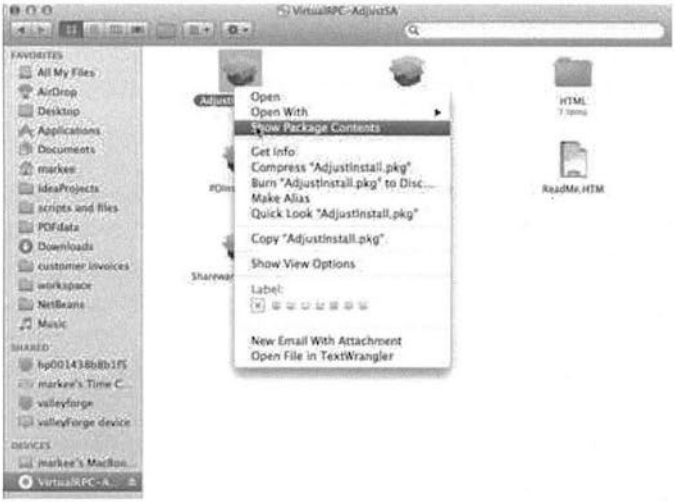
Mac Matters

Mark Stephens

A practical item this issue to help you make the most of your Mac, so straight in.

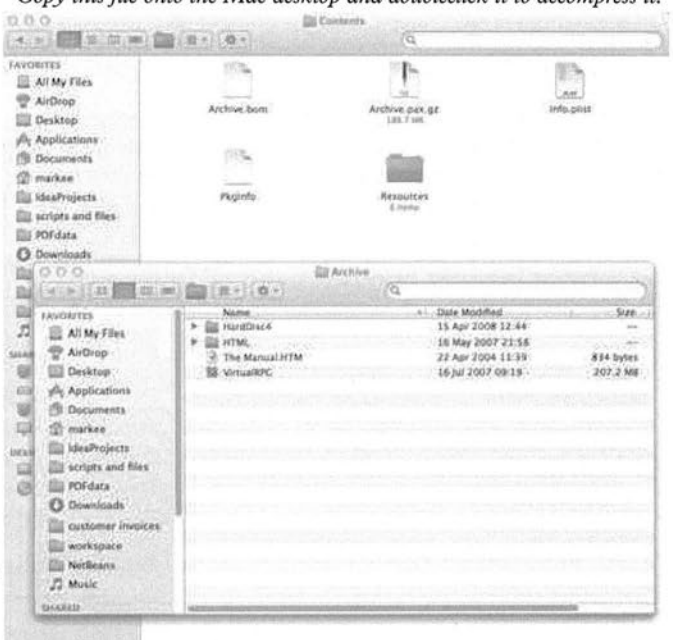
Harddisk4 for VirtualRPC

I had to recreate my VirtualRPC setup for my new Mac laptop and had a nasty shock – the installer would not run. In the past I just reran it to create a new "vanilla" copy of Harddisk4. The good news is that you can manually extract it still – these two screenshots show how.



1 *Load the RiscPC-Adjust CD and open it. Instead of doubleclicking on the application, from its menu choose the option* Show package contents.

2 *Inside is a Contents directory and inside that is the file* Archive.pax.gx. *Copy this file onto the Mac desktop and doubleclick it to decompress it.*



RPCEmu rivalling Virtual Acorn

An alternative to the commercial emulators from Virtual Acorn (VirtualRPC for Mac costs £79 plus postage) is RPCEmu (free). This free upstart used to be a bit fiddly to set up, but the team at Rool have put together a USB memory stick (£10 plus postage) containing a ready-to-go version with all the rough edges tidied up.

First impressions of RPCEmu on my Mac look very good. It lacks some of the polish and features of VirtualRPC: for instance, networking is complex to set up, and it does not have a full-screen mode. On the

plus side, besides being much cheaper, RPCEmu allows you to run RISC OS 5 – which is where development seems to be happening at the moment.

Even if you already have VirtualRPC, the free emulator is worth a try. You can even share directories between both. More on that and more feedback on RPCEmu next time.

Hey! Back up!

This is a reminder to me about the importance of backups. The Mac has an automated backup system called Time Machine. (You have remembered to enable it, haven't you?) But it is also worth making a snapshot of the Harddisc4 folder every so often.

If you select your Harddisc4 folder once and then from its menu choose the Compress option, you can make a zip copy. You have a backup copy now!

Mark Stephens
markstephens@idrsolutions.com

Mark was asked to do a technical talk at a PDF conference in Belgium. His talks are renowned for the mix of complex facts and kitten pictures.

hint&tip Redirecting output – *spool without gobbledygook

In *Archive 23:1* I described some methods of listing directory contents into a file; I suggested using `*Spool` to direct output from a command such as `*Cat` to a file. I admitted that, if run from an obeyfile while in the desktop, the file would contain unwanted bytes. I ended by saying there must be countless ways of swinging that particular `*Cat`.

Well, two suggestions have emerged:

1 Use filetype `taskobey` (&FD7) for the obeyfile. Then the spooled output does not contain the screen-control bytes, though a taskwindow is opened that shows the output, and the line-endings in the file are LF+CR instead of the usual Unix or RISC OS convention of LF only.

2 Julian Fry (*julianfry@orpheusmail.co.uk*) pointed out that the commands in an obeyfile can include redirection. Thus, instead of:

```
*Spool Dirlist
*Cat <directory>
*Spool
```

one need merely have this (braces and spaces are vital):

```
*Cat <directory> { > Dirlist }
```

albeit this too gives LF+CR line-endings. In both examples, the file called Dirlist will be created or rewritten in the current directory unless qualified with a path.

Output redirection, along with output appending and input redirection, are described on page 381 of the RISC OS 3 User Guide, and should be better known (by me at least).

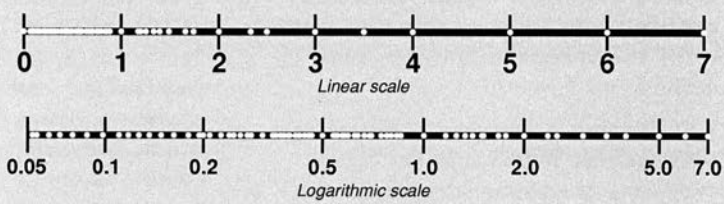
Bernard Boase *b.boase@bcs.org*

Numbers game Part 6 of 7

Keith Hopper

Take another look at those two lines marked with the points corresponding to the *four*+*four*-bit rational number domain from the previous article *(Archive 23:4)*, reproduced here for ease of reading.

The only numbers that can be represented in a four-bit computer



Did you work out why the logarithmic line shows a cluster around 0.5 (well, actually approximately 0.47)? The answer lies in the ranges for numerator and denominator. which centre on [7, 15]. If the two had been in the same range then the clustering would have been about 1. Just keep these thoughts in mind as clustering will be revisited at the end of this article.

Modelling

To the purist all that we have been doing with this *four*-bit microprocessor scheme for representing rational numbers is to model a tiny subset of the mathematical rational-number domain. Computer representations of numbers fall into the same category as "zero.succ", "minus-one", "forty-second", etc which were used in earlier articles in this series – just names for numbers. Mathematicians would call the set of bit-patterns in computer memory by the name *model numbers*. These are the only valid numbers for which the computer can calculate answers.

Think in the mathematical domain of rational-number values for the purposes of obtaining a result, but there is then a need to convert any such result into a model number for storage in the computer. The table below shows a few (more or less) randomly chosen calculations for our *four*-bit machine, their results, the nearest model number – and the resulting error (if any).

So. Out of eight random expressions one is correct, two overflow and the others have an error of up to [1, 3] – not exactly a very good model from an arithmetic point of view!

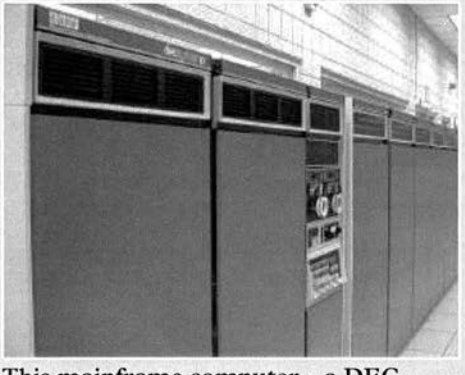
Expression	*Answer*	*Model Nº*	*Error*	*Notes*
[2, 1] + [2, 1]	[4, 1]	–	*overflow*	which *might* be detected
[4, 3] + [4, 3]	[8, 3]	[3, 1]	[1, 3]	
[4, 15] × [3, 2]	[2, 5]	[2, 5]	[0, 1]	
[4, 15] + [3, 2]	[53, 30]	[2, 1]	[7, 30]	note symmetry
[3, 2] – [4, 15]	[37, 30]	[1, 1]	[7, 30]	
[3, 7] – [2, 13]	[25, 91]	[3, 11]	[–2, 1001]	
[2, 3] / [1, 7]	[14, 3]	–	*overflow*	which *might* be detected
[2, 3] / [3, 7]	[14, 9]	[3, 2]	[–1, 18]	

Redundancy

This rational model also, you will remember, incorporates about a third non-normalized values which, from a computer point of view is rather inefficient. In the days when 8- and 16-bit computers were widely used a similar although rather more efficient bit-pattern allocation was used over a greater range of values. Redundancy is still present here, however, so a more efficient model was urgently needed if arithmetic was not to be very restricted where non-integral results were likely to occur.

The situation is, however, rather worse than has been hinted at so far. When evaluating an arithmetic expression such as those in the table above, the computer's arithmetic logic unit (ALU) had to be able to determine which was the nearest model bit-pattern. The only way that this could possibly be done is by the provision of *more* than four-bit calculations in the ALU in order that rounding up or down could be decided – before normalizing the result!

A mainframe with 36-bit words



This mainframe computer – a DEC System-10 made by the Digital Equipment Corporation in the late 1960s – was unusual in having 36-bit words that could contain five seven-bit-encoded characters, rather than the now almost universal encodings which are multiples of eight bits. Arithmetic was carried out on entire words or, optionally, half words (18-bit).

The "console" is actually another DEC computer – the 16-bit PDP-11.

Plethora of models

In the early days of computing, every manufacturer of however many bits his processor handled in registers – 4, 8, 12, 16, 18 and 24-bit processors were made before the (8, 16, 32, 64)-bit series was standardized – although one outstanding exception, shown in the box, used 36-bit words. These early machines all developed different numeric models to handle non-integer arithmetic. Most of those were some variant equivalent to the written scientific exponential notation. Few of these early ideas produced the same answers as any of their rivals' machines (shades of the remainder-and-modulus problem in Python and PHP encountered in the previous article). Something just had to be done to enable results obtained on one machine to be duplicated on some other using the same set of data.

Scientific notation

So-called scientific notation was introduced in written calculations to save a lot of writing of zeros – before or after the decimal point. Thus 1,000,000,000 would be written as 10^9 or, avoiding the superscript, as just "10e9" *[e for exponent —Ed]*. Similarly −0.0000000002 would be written as -2×10^{-10}.

The use of base 10 numbers was chosen because decimal digits are used in the written representation of values. It wasn't long into the "computer era" before even shorter shorthand notation came into use: `1e9` for 10^9 and `-2e-10` for -2×10^{-10}.

These simple examples have only a single significant digit. However, where more significant digits are available, a decimal point appears after the first digit and the exponent value is altered accordingly. Thus the value 4321 would be written as `4.231e3` and the value 0.004231 as `4.231e-3`.

Before diving further into approximate number systems, it is appropriate to name the four (+1) components of this written notation:

(+)	4.231	(e)	(–)	3
sign (optional)	mantissa		exponent sign (optional)	exponent value

The `e` for the exponent is redundant when the number is represented as a bit-pattern in a computer – though it is needed for readability on paper or screen.

Approximate models

Remember that, in the absence of an infinite supply of memory (or paper), all arithmetic which is not integral just has to be approximate!

So. How does one go about defining an approximate number domain? The mathematics behind such a model may be advanced, but understanding what has to be done is really quite straightforward. Look again at this number:

+	4.231	–	3
sign	mantissa *<length>*	e-sign	exponent *<length>*

- The two signs have to be there, in a computer at least, so storage (or paper) is needed for them.
- The mantissa (m) has a certain number of digits which indicates the precision (p) which the approximation provides.
- The exponent has a length too. This indicates the largest and the smallest exponent value which can be used. In practice and in theory too there should really be two values here – one (*emin*) giving the smallest fraction exponent, the other (*emax*) giving the largest whole exponent.

However, one tacit assumption is hidden in the last paragraph: that the number base is 10 – since decimal digits are used for representation. Inside a computer it is quite likely that some other base will be used (possibly 2, 4, 8, even 16). Since we don't really care about that for the moment, let me use the mathematician's word "radix" (r).

Instead of writing 10^{42}, therefore, I will use r^{42}. This means that we can write an arbitrary number in our model as something like this:

$m \times r^e$ where m is the mantissa
and e is the exponent
and r is the radix (or base)

instead of writing

4.231×10^{-3}

(m in this case is 4.231, r is 10, e is −3).

The question of precision

I hope this is reasonably straightforward to understand, as we need to go one step further before looking at the results in our *four+four*-bit system.

You will, I am sure, have noticed that so far the precision has not been taken into account – and that I have gaily inserted a decimal point into the mantissa. This was done in the hope of avoiding confusion when you are getting to grips with these new ideas. The number written as 4.231 should really be written *without* the decimal point.

Let us presume that our precision is five decimal digits – just for the purposes of illustration. This gives the number as:

$$
\begin{array}{lll}
 & 4.2310 & \\
\text{or} & 42.310 & \times 10^{-1} \\
\text{or} & 423.10 & \times 10^{-2} \\
\text{or} & 4231.0 & \times 10^{-3} \\
\text{or} & 42310 & \times 10^{-4}
\end{array}
$$

The mantissa is now a whole number (in the integer domain, $\mathbb{Z}$) with five digits of precision – so our original number now becomes

$$
\begin{array}{ll}
 & 42310 \times 10^{-7} \\
\text{or} & 0.0042310 \text{ as originally written.}
\end{array}
$$

Our final version of the number has now a larger negative exponent: −7. Somehow or other this has had to be changed to take account of the requirement that the mantissa had to be written without a decimal point. Look at it again:

43210×10^{-7}
(p=5)

What would have to happen if we decide to increase the precision to 6, say? Look:

$43210 \times 10^{-7} \equiv 432100 \times 10^{-8}$
(p=5) (p=6)

The exponent has had to go down to −8 to take care of the extra digit. The value, however, has not changed:

432100×10^{-8}
= 0.004321 (as before).

A yet closer look

We need to take another look at this point. The version above has a whole-number mantissa. This could quite happily be replaced by a pure-fraction mantissa (as long as we know which one we are dealing with), thus:

0.432100×10^{-2}
= 0.004321 (as before).

Note carefully the difference in these two versions:

$432100(.) \times 10^{-8}$ and $(.)432100 \times 10^{-2}$

This difference is only in the exponent of the numbers in the model. This means that the algebraic version of the number should really be given as:

$$m \times r^{e-p}$$

where m, the mantissa, is 432100
r, the radix (base), is 10
e, the exponent, is −2
p, the precision, is 6.

Note here that the precision is measured in the number of digits in the base number system (which just happens to be decimal in the example). The final point to note is that the mantissa is always between

10^5 (which is r^{p-1}) and $10^6 - 1$ ($r^p - 1$)

– in other words, between 10000 and 99999 in our example. Providing that a number in our approximate system is either "zero" or obeys the restriction given, the number is said to be *normalized*. We have already met this term when considering rational numbers – although in that case it was defined in rather a different way. The definition here is to ensure that "full precision" will always be given to the results of arithmetic operations.

It is not unknown, by the way, for practical

IEEE standards 754 and 854

The US Institution of Electrical and Electronic Engineers first developed a standard recommendation for approximate arithmetic in 1985 to persuade manufacturers to make sure their machines did arithmetic using the same model as other manufacturers. The 2008 version of this standard also includes an earlier standard (854) which defined radix-independent arithmetic.

Several hardware options are permitted to manufacturers, although the arithmetic results are defined to be the same in all cases.

hardware permitting the use of non-normalized numbers – which get very small indeed but need some sort of indicator to say they are non-normalized.

The four-bit illustration again

Before going on to look at approximate arithmetic, it is worth looking again at our 4+4-bit number system, this time using four bits for the mantissa and four for the exponent – both being signed. Remember that this is purely for illustration purpose of the ideas behind this different model, with no intent to offer a practical mechanism.

The signed mantissa has only three bits devoted to value. Remembering that the leading bit always has to be a 1, the only possible bit-patterns are "100", "101", "110" and "111" – yes, only four possible signed binary *normalized* values in the exponent notation.

The exponent can, of course, have any signed value between −7 and +7. Doing a bit of arithmetic, this means there are a total of 121 model numbers – including zero. This is even fewer values than for rational numbers. They are, however, evenly spaced in each exponent range. Turning them into decimal equivalents for ease of reading gives each range with values 4, 5, 6, 7 multiplied by anything from 2^{-7} to 2^{+7}.

Just to take one value as an example, think of the integer numbers which are represented thus:

1 becomes 4×2^{-2}
2 becomes 4×2^{-1}
3 becomes 6×2^{-1}
4 becomes 4×2^{0}
5 becomes 5×2^{0}
6 becomes 6×2^{0}
7 becomes 7×2^{0}
8 becomes 4×2^{1}
etc

which all seems to work as might be expected. What numbers, however, between decimal 1 and 2 can there be in this model?

We have 1.25 becomes 5×2^{-2}
1.50 becomes 6×2^{-2}
1.75 becomes 7×2^{-2}

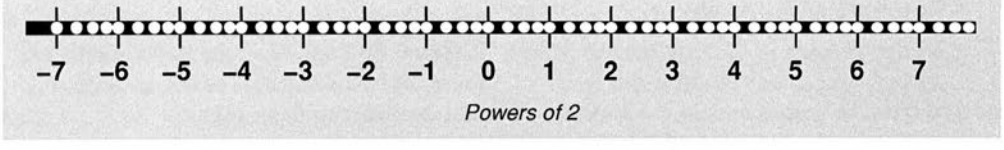
So what about the range between decimal 2 and 4?

We have 2.5 becomes 5×2^{-1}
3 becomes 6×2^{-1}
3.5 becomes 7×2^{-1}

all beautifully (more or less) evenly spread from one power of two to the next. Convince yourself by checking the values between a half and one. The even-handed division of each exponent range is obviously more practically useful – despite the smaller number of normalized numbers than the rational case.

Is this better?

The figure below is a revamped version of the *logarithmic* number bar with which this article started out. Note that the largest value is 224 (in other words [224, 1]), and the smallest is just 1/128 (in other words [1, 128]) with just 58 values in between. Just compare the two and see what you think. Is mantissa-and-exponent



Powers of 2

representation better – or not? It all depends on what you mean by better!

... and now

I seem to have run out of space and at the same time reached an interesting point in the discussion of approximate numbers. The question I leave you with for the final instalment is this:

"How on earth can one actually work out the result of, say, some addition in a practical piece of computer hardware?"

Keith Hopper

Keith was involved with computers ever since students helped build Edsac II in the Cambridge labs in the mid-1950s. He taught computer science at Leeds, then at Waikato in NZ. He died in 2011.

Compiling with GCC Part 20

Rob Johnson

In the last few parts of this series I've been exploring some obscure differences between Acorn/Castle's **!ObjAsm**, the internal assembler **as** in GCC versions up to 3.4.6, and the standalone assembler **asasm** provided in 4.1.1 Release 2. All these are theoretically identical, but they have some mostly accidental differences.

GCC's release 4.1.1 was a marked departure from previous ones. They no longer wanted to spend valuable resources converting certain peripheral aspects of the compilation system from Linux to RISC OS format, and instead concentrate on the language facilities of C and C++. They stopped translating from Linux's native ELF files to Acorn's proprietary formats. And they used Linux GCC's assembler rather than an Acorn-style one.

Now I will turn my attention to the internal as in all 4.1.1 releases. I will take a tip from GCC's documentation and refer to it as gas (GCC as). The old as was written outside of GCC to simulate !ObjAsm, and co-opted by GCC to fill a hole. 4.1.1's asasm is a new version of that.

The gas assembler

The new gas is totally different from the other assemblers. Thus the source code it expects is quite different too.

The RISC OS version of GCC is ported from the Linux original, and the new gas is ported from the assembler in the Linux version. That assembler is designed to work on many machine types. It doesn't expect a particular machine-code instruction set. The instruction set of the target machine must be plugged in. Thus the heart of the source code is (almost?) the same as for the other RISC OS assemblers.

But the peripheral things like comments, labels and assembler directives are different from the other RISC OS assemblers. The Linux assembler has its own standard, which it expects to be (basically) the same on all machines.

Documentation

gas has copious documentation supplied with !GCC. Acorn/Castle provided a manual for !ObjAsm (and other assembler tools and topics). asasm and the old as don't have their own documentation, because they are supposed to mimic !ObjAsm. So I will base my experiments on these two available sets of documentation, and explore out from there.

Sample code

In my previous articles I used a Rool assembler source called modulewrap to generate interesting things to investigate. But I didn't supply modulewrap for you to play with, because it required lots of other Rool files to make it assemble properly. Instead I extracted the code from modulewrap and its environment that was relevant to a particular problem, and turned it into simple example code that I could present to you.

It is even more true that I can't use modulewrap itself to probe gas. The differences in style between modulewrap and a gas source are so great that it would be a Herculean effort to modify it. And I'd have to do the same thing to every subsidiary source that modulewrap drags in. So I will have to find other simpler sources to use. Or invent them myself.

Comments

The first thing I need to sort out is the way of writing comments in the code. Not because it is the most important part of code, but so that I can put comments in my other example sources.

The documentation indicates that all the assemblers accept the same kind of comment line. A comment is introduced by a designated symbol. If the line starts with the symbol then the whole line is a comment. If the symbol appears partway along a line then the rest of the line is a comment.

The gas documentation says that the Linux assembler will accept on various machines ;, @, #, !, | or -- as comment indicators. For Arm machines it's @. !ObjAsm's manual says it uses ;. One can safely presume that asasm and the old as follow in !ObjAsm's wake. This immediately raises the question: If the Linux assembler can accept ;, why didn't our GCC decide to use the same symbol as the other Risc os assemblers? (We'll see the answer to that in the section called "Multi-line lines".)

My extremely simple file Comment tests this out. Just change the comment symbol to what you want and see what will be rejected. (The following paragraphs will suggest why I don't just put every possible comment symbol in the file at the same time.) (Note that some assemblers use some of these possible comment symbols for other purposes, and the error messages will reflect that.)

As expected asasm and old as accept only ; as a comment indicator. And it is true that !ObjAsm uses ; and gas uses @. But things get stranger after that.

Undocumented comment symbols

By the time we've run all the possibilities through all the assemblers we find that gas also accepts #. And !ObjAsm can handle all three of ;, @ and #.

To explore this a bit more, I now move on to the file Comments, which contains three hopeful comment lines, starting with either ;, @ or #. Running it through asasm and the old as confirms that they like only ; comments. And that gas accepts @ and #. Significantly it accepts both in the same file. And shuffling the contents of the file shows that it doesn't matter what order the ;, @ and # lines appear in. Which sounds obvious until you try it on !ObjAsm.

There it transpires that !ObjAsm will accept ; comments or combinations of @ and # lines, but not both types in the same file. The choice depends on which comment symbol it sees first. In other words, !ObjAsm is built to cater for its official comments style (à la asasm and old as) or

alternatively the Linux version.

This reminds me of the fact that the Acorn/Castle compilation system works with Linux ELF files (as seen in GCC 4.1.1) as well as Acorn's own filetypes (as used by earlier GCC versions). I wonder how far Acorn/Castle's acceptance of Linux's assembler style will extend?

Double-comment system

Why does gas have both @ and # as comment indicators? Its documentation offers a clue when it says the some machine types use two comment symbols and treat them differently. One of them is valid only if it starts the line, while the other one can also be used after code on the line.

The file DoubleComment containing

```
@ Comment
 MOV PC,R14 @ Comment
# Comment
 MOV PC,R14 # Comment
```

shows that gas does use @ and # as a double-comment system. The fourth line is rejected, indicating that # can be used only for a line that is all comment. @ can do that and also be preceded by code.

Before continuing I want to clarify some things about what can share a line with a comment. When I say that # can't have anything before it on the line, I don't count spaces (or other things deemed whitespace character, such as tabs). In order to qualify as a line that is only partly comment, the early part of the line must contain some meaningful characters. And they must make sense to the assembler else the test will be confused by extra error messages. Thus I had to put in an actual machine-code statement like `MOV PC,R14`, because other things such as assembler directives would be different for different assemblers.

My next question was whether the double-comment system applies to !ObjAsm too. And it does. But again things don't end there.

Schizophrenic !ObjAsm

If I write another file CommentCode with similar lines using just the ; symbol

```
; Comment
 MOV PC,R14 ; Comment
```

and pass them to the old assemblers, a strange thing happens. They all, including !ObjAsm which accepted the @ # version, complain that there is a missing `AREA` statement. If I add an `AREA` statement, then asasm and old as assemble OK, but !ObjAsm goes into a loop, which can be avoided only by adding a terminating `END` statement.

The final version of CommentCode is

```
; Comment
 AREA jim, CODE
 MOV PC,R14 ; Comment
 END
```

This is all valid according to the !ObjAsm manual. Code lines need a designated code area to put the object code in. A source file must finish with `END` (although asasm and old as obviously ignore that requirement).

But the more interesting thing is that when presented with a file containing @ or # comments (or both), !ObjAsm didn't recognize `AREA` and `END`. But if I remove them it seems to complain that there's something wrong with a default area. As we will see later, gas has a different style of directives, with different names. But !ObjAsm doesn't recognize those either.

The conclusion that I am directed towards

is that !ObjAsm accepts two styles of source. One is the official Acorn/Castle version. The other uses the same comment style as the Linux gas version, but an unidentified directives system. The choice is determined by the style of the first comment line the assembler comes across. (If it reaches code before any comments, then it defaults to the Acorn/Castle style.)

C-style comments

Before leaving comments I must mention that the gas documentation also says that comments deliminated by `/* ... */` will be accepted. These are comments in the style used by C. As in C, such a comment can extend over multiple lines. The additional C++ one-line comments prefixed by // are not allowed. File CComments shows this is true for gas.

As would be expected, asasm and old as can't handle this. But also I find that !ObjAsm won't accept such comments either, even if forced into Linux mode by an `@` comment.

Code lines

The file CodeLineGas includes

```
MOV PC,R14
```

This is a line of RISC OS assembler code. As suggested above this line by itself is acceptable as a source file by gas, and will assemble successfully.

But asasm and other Acorn-based assemblers need more. File CodeLineAsasm contains

```
AREA jim, CODE
MOV PC,R14
END
```

As mentioned already an `AREA...CODE` line is needed to specify where to put the following code. And !ObjAsm also insists on `END` to terminate the source.

This file successfully assembles under !ObjAsm, asasm and old as.

All the !ObjAsm code lines start with a space. That is because the Acorn syntax for a code line is

```
label instruction ;comment
```

with all the parts optional.

The characters `MOV PC,R14` constitute an instruction. The preceding space is so that `MOV` isn't taken as a label. The same applies to the other two lines, even though they aren't machine-code instructions.

Labels

gas also supports the three optional parts of a line, dividing it into label, instruction and comment. As with the other compilers, the comment is recognized by its initial defining symbol.

As we have seen, the only way the Acorn compilers recognize a label is that it starts in the first character of the line, as demonstrated in the file LabelAsasm, which contains

```
 AREA jim, CODE
something MOV PC,R14
 END
```

But gas doesn't have this restriction, because its labels have a special trailing symbol :, as seen in LabelGas

```
 something: MOV PC,R14
```

As stated in the previous section, this also allows gas to start its instruction right at the beginning of the line if there is no label. In other words, gas has no positional restrictions apart from the order of the optional fields.

The defining : also allows another difference in gas. The file MultiLabelGas

containing

```
something: else: MOV PC,R14
```

shows that gas supports multiple labels on the same line. The Acorn system which recognizes a label only because it starts at the beginning of a line obviously can't do this.

There is another example of !ObjAsm's schizophrenia here. It doesn't like the `label:` syntax in normal mode, but insists on it in `@` mode.

Multi-line lines

Another extra facility in gas is the ability to have more than one instruction on a line. The separator between instructions is `;`. (This explains why our version of gas doesn't choose to use Acorn's `;` as comment indicator.)

The file MultiLineGas shows more than that,

```
something: MOV PC,R14 ;else: MOV PC,R14
@comment
```

(This is a single line.) Each instruction can have its own preceding label(s). But comments can occur only at the end of the whole physical line. If you put a comment after the first instruction you'll find the second one is ignored – as you can prove by making it an illegal one.

The Acorn system has no equivalent scheme. It's difficult to tell whether !ObjAsm can handle multi-line lines in `@` mode because the area problem gets in the way.

Directives

gas also has a special symbol to indicate directives: they start with a ".". The Acorn system doesn't use such a symbol. Instead it recognizes all directives by their position and names.

Acorn assemblers treat directives like instructions, i.e. they occupy the middle slot in the trio of optional line elements. That means they must be preceded by at least a whitespace character. By contrast, gas directives are not so-constrained.

We have already met the Acorn `END` directive. gas has the equivalent `.end`. However, experiment has shown me that gas does not insist on it, but if present it will make gas ignore any lines that may follow it.

Any parameters must follow an Acorn directive. At least one RISC OS gas directive takes advantage of its non-positional nature to have a paramenter in front (`foo .req r0` gives register `r0` an alias `foo`).

Areas and sections

Assemblers divide their output into regions with different properties. The main division is usually between code and data.

Acorn calls the regions areas, and they are created by the `AREA` directive. gas calls them sections, which can be defined by the `.section` directive. Lines that follow one of these directives will be put in that region.

gas goes even further. While Acorn assemblers insist that you define all areas from scratch – for instance `AREA jim, CODE` or `AREA fred, DATA` – but gas provides predefined sections `text` and `data`. These sections can be invoked as directives by `.text` and `.data`.

File SectionsGas contains

```
.text
MOV PC,R14
.data
.string "string"
.text
MOV PC,R14
```

This shows some code followed by some data and then some more code.

Similar lines in AreasAsasm are

```
AREA jim, CODE
MOV PC,R14
AREA fred, DATA
DCB "string"
AREA alf, CODE
MOV PC,R14
END
```

This points up another slight difference. Whereas in gas I can use `.text` twice to put stuff in the same section, in the Acorn example I have to give the second `CODE` area a different name.

There is another difference that we have seen earlier, in that you don't need the initial `.text`, because gas assumes lines are code unless told otherwise.

The results I believe are basically the same. The Acorn system combines all areas with the same definition (in this case `CODE`) together. The gas system combines all `text` sections together.

There are more properties than just `CODE` and `DATA` that can be specified for an Acorn area. There are also more types of gas section than those represented by `text` and `data`. You can define such other types of section via `.section name, properties`.

When you define such a section the immediately following lines of code will be placed in that section. But creating such a `name` section doesn't generate a directive `.name` for reuse further down the source. Like Acorn you have to redeclare a section of the relevant type. But Sections2Gas

```
.section jim, "x"
MOV PC,R14
.data
.string "string"
.section jim
MOV PC,R14
```

shows that you can reuse the name, and it will carry over the properties.

Files

All the source files mentioned above can be downloaded from the Archive website.

Files: see p.2

I would also advise you to add the `-c` parameter to the `gcc` command in the Compile_s file in all copies of !GCC.bin. This will make all versions of as and gas stop before linking. My examples are not runnable programs.

Summary

I think I've covered all the basic differences between gas and the Acorn assemblers. Conversion between the two forms could probably be automated in a sequence of Edit commands or a StrongEd script.

Next time

I'm not sure. I see three possible strands of continuation. Explore more gas facilities. See what gas does in the anomalous situations I described in recent issues. Develop an automated conversion script. Or like Monty Python's Spanish Inquisition add a fourth option and do something completely different.

Rob Johnson *robfj@robfj.bbmax.co.uk*

I'm approaching the need for a cataract operation. And I'm exploring what options I may have within that.

The secret life of algorithms Part 7

Nicholas Cutler

This part of the series will return to looking at quite simple algorithms. This time the algorithm is for a commonplace task – which I understand occupied another reader some years ago (see Feedback, *Archive 22:11*) – to find the square root of a number.

Widely used

Perhaps this will, initially, seem to be another example of a fairly specialized algorithm. But bear in mind that, even if you do not directly calculate square roots yourself, the function is nevertheless used frequently in computer software. For example, a drawing program will need to calculate a square root when determining the length of a diagonal line. Furthermore, it seems appropriate to discuss the problem here, as numerical computation in the sciences was the first application of computers.

It may also seem surprising that for such a simply stated problem, there should be so many approaches to its solution. This is in part because the numbers we need to work with may vary greatly in size. An efficient method on our 32-bit machines may not work so well with 32,000-digit numbers! Secondly, there has been a need to calculate square roots for several centuries, and methods have evolved to suit the means available. Of the two algorithms I will discuss here, one is based on a pencil-and-paper method – which may be familiar to readers who can remember school arithmetic before calculators.

The problem, of course, needs little introduction: namely for a given x, finding the number y, such that $y^2=x$, or more intuitively performing the inverse of squaring a number.

This is such a fundamental operation that most processor architectures that support hardware floating-point operations include a dedicated square-root instruction. However, at some point, whether in hardware or software, this function will need to be calculated using simpler operations, and that is the focus of the present article.

Throughout this discussion I will assume that x is positive to avoid introducing complex numbers. Similarly, while the methods presented here are accurate to within the limitations of the computer's arithmetic, the square root of some numbers is irrational, and I have further avoided infinite fractions for the sake of simplicity.

The old school method

Having stated the problem, it is then natural to start by examining the familiar manual method and trying to adapt it for use on computers. As you will see (or recall!) from the example below, the calculation is set out like a long division:

```
  1.7 3 2   <-- Square root
√03.000000  <-- Operand
 03
 01
 ---
 02 00
 01 89
 -----
    1100
    1029
    ----
     7100
     6924
```

At each step in the algorithm, we proceed

to "bring down" the next two digits from the operand, and set them alongside any remainder from the previous stage. Denote this quantity by c.

Now let p be the digits of the root which have been found so far (for the first step this is zero). Then find the largest single digit, x, such that

$$y = (20p+x)x$$

does not exceed c, and write this value up as the next digit of the root.

Now subtract y from c to form a new remainder. Stop when the remainder is zero, or the root has been found to sufficient accuracy, otherwise repeat the whole procedure.

What each stage of the above does is to find the digit which will make $(10p+x)^2$ as close to the operand as possible without exceeding it. Expanding this expression gives:

$$(10p+x)^2 = 100p^2 + 20px + x^2$$

which shows the derivation of y.

It's better in binary

The good news is that this method becomes much simpler when implemented in binary, making a very efficient algorithm for the computer. The basic idea is to start with the value of p set to zero, as in the decimal calculations above, and then find the digit x which when added in the place immediately to the right of p makes $(p+x)^2$ less than or equal to the operand.

This is very much like the procedure in base 10, but with the huge simplification that the digit x is either 0 or 1. Furthermore, in binary, most of the necessary values can be updated incrementally using only simple integer operations like addition and bit-shifts – instructions that all microprocessors support natively. This algorithm, which is known as the **restoring method**, can therefore be expressed in Basic as follows:

```
FOR I% = 1 TO 10
PRINT FNsqroot(I%*1E4)
NEXT
END
:
DEFFNsqroot(op%)
LOCAL res%,one%
res% = 0
one% = 1<<30
WHILE one% > op%
one% = one%>>2
ENDWHILE
WHILE one% <> 0
IF op% >= (res% + one%) THEN
op%-=(res% + one%)
res% = (res%>>1) + one%
ELSE
res% = res%>>1
ENDIF
one% = one%>>2
ENDWHILE
=res%
```

In this procedure, note that `res%` is the partial result, and `one%` is equivalent to the digit x in the above descriptions. In each iteration of the main loop, we then test to see whether this can be added to the partial result, before updating the variables with simple shift operations.

Another point to note is that, for integer numbers at least, the result cannot possibly be greater than the operand, and indeed will generally have half the number of significant figures. Thus, an initial stage in the above program is to scale the `one%` variable. In an efficient assembly-language implementation this will often be done by counting the number of leading zeros in

the binary representation of the operand. Many processors now offer a "count leading zeros" instruction to do this quickly (including version 5 and above of the Arm architecture).

Ah, the snag

There is, however, one problem with this simple approach, namely that the square root of a number is rarely an integer. Accordingly, to rectify this shortcoming a practical implementation will have to use floating-point arithmetic (the usual representation of fractional numbers in most programming languages). This option is only a little more involved.

First, split the number into its mantissa (the fractional part) and exponent so that $x = m \times 2^p$. Then note that

$$\sqrt{(m \times 2^p)} = 2^{p/2} \sqrt{m}$$

so that it is sufficient to halve the exponent, and then calculate a single square root of the mantissa. Where the exponent is odd, it is normal to round down when halving it, and then multiply the mantissa by two.

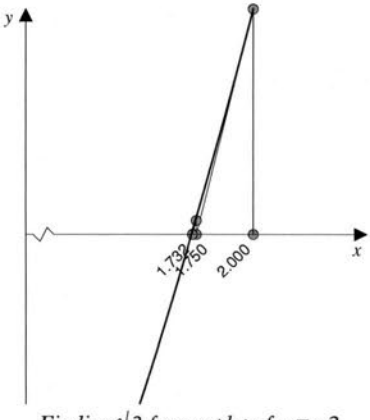
As a point of interest, the floating-point emulator in RISC OS machines uses this algorithm, together with the above variation, when calculating square roots. As with the above version in Basic, the FPE's implementation in assembly language uses only simple instructions that the Arm can carry out very quickly. Ultimately this makes the square root no more difficult to program than the division operation.

Bigger numbers need Newton

The ease of implementing this method using only integer arithmetic is its major attraction. However, in the worst case, it requires one iteration for each bit in the result. As numbers become larger it becomes preferable to sacrifice simplicity and use a different method with faster convergence. Most libraries for handling large numbers to arbitrary precision use Newton's method for this purpose. Here's how this algorithm works.

Strictly speaking, the **Newton-Raphson iteration** is a method for solving (that is, finding roots of) arbitrary equations. By definition, this iteration takes an initial guess for a root, then produces a sequence of progressively more accurate solutions. We can keep going for as long as necessary to produce the precision we want.

More formally, starting with an initial appoximation to $f(x) = 0$, denoted by x_1, Newton's method is applied to produce a more accurate solution x_2. At each subsequent step, a new x_{n+1} is calculated from x_n. In this particular case, we are interested in solving the function $f(x) = x^2 - a$, but keep in mind that the method is much more general.



Finding √3 from a plot of y = x2

To see how this method works, it is easiest to consider a graphical interpretation. The diagram shows a plot of part of $y = x^2$ and the first few steps in finding $\sqrt{3}$.

Starting with an initial estimate of $x_1 = 2$, we consider the corresponding point on the curve $y = x^2 - 3$. From this point, draw

a tangent (a line parallel to the curve) and note where it crosses the x-axis; this intersection then becomes the new approximation.

This procedure can be repeated as often as necessary, although in practice the results converge rapidly. Indeed, the above diagram is able to show only one further step before the lines become too close together to see!

We need the tangent

To develop the method to something suitable for use on a computer, it is necessary to represent the intuitive procedure of "draw the tangent" with more precise operations. We already know one point which this tangent passes through, so to uniquely specify the line we need to know its slope as well. This slope is, of course, the same as the gradient of the curve at the point in question – a quantity which is continuously changing.

If the curve were approximated with a series of very short straight line segments (which is effectively how the computer would draw it), then the gradient would simply be the slope of the relevant line segment. Now imagine that the approximating segments were made ever shorter, so that, eventually, we would finish up with the precise value of the gradient at one point. The gradient of this infinitesimally small "segment" is simply the derivative of $f(x)$, which is denoted by $f'(x)$. Thus, it is now possible to express the new value of x_{n+1} is terms of the previous x_n as:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

For the case in which we are interested, the derivative $f'(x)$ is simply $2x$. For readers unfamiliar with calculus it is sufficient to understand that this is the gradient of $x^2 - a$ at a given point. Putting these values in for the function and its derivative, and then simplifying the result:

$$x_{n+1} = \frac{1}{2}\left(x_n + \frac{a}{x_n}\right)$$

Babylon meets Basic

This specific iteration was known long before Newton's time, possibly since the Babylonians. It is still a very attractive algorithm for calculating square roots. On the computer this method can be implemented very simply, using Basic's regular floating-point arithmetic:

```
DEFFNsqr_newton(op)
LOCAL oldx,newx,eps
oldx = FNapprox(op)
REPEAT
PRINT oldx
newx = (oldx + op/oldx)/2
eps = ABS(oldx - newx)
oldx = newx
UNTIL eps < 1E-10
=oldx
```

As you can see, implementing Newton's iteration takes only a few lines of code; the main difficulty is finding an initial approximation. In the complete version of this program, I have simply used a few iterations of the restoring method, with integer arithmetic as before. Another effective way of finding a first guess is to use a polynomial approximation. Once again, in the interests of simplicity I am assuming that the iteration will converge for most reasonable starting values. In reality, it is possible to be much more precise and, except in the most trivial cases, it is unwise to use numerical methods blindly.

➥Files: see p.2

When trying this program out, it is useful to use Basic64 to gain extra precision in the arithmetic. Doing this will show that the number of correct digits in the intermediate results nearly doubles at each stage. Indeed, on small examples, it converges so quickly that it can be difficult to tell!

Clearly then, this algorithm requires many fewer iterations than the restoring method, but with the disadvantage of requiring floating-point operations. This tends to limit its use to those cases when large, arbitary-precision numbers are needed, and where the faster convergence is of greater importance. Conversely, if floating-point calculations can be performed in hardware, then Newton's method becomes attractive even for relatively small numbers.

Anything better yet?

The two methods discussed above both have their uses, but are they the last word in calculating square roots? The answer is "nearly, but probably not!" It was only a matter of time before people started competing to calculate thousands or even millions of digits of irrational numbers such as $\sqrt{2}$. Even Newton's method can be improved upon.

A relatively new algorithm used in the Gnu MP (multiple precision) library, well known in the Linux world, is a recursive scheme that repeatedly divides a large number into two parts. Despite this, it is only slightly faster, and part of that speed depends on an unusual way of multiplying large numbers:

Normally multiplication of large numbers is carried out in a way analagous to the usual pencil-and-paper method for long multiplication. It is possible to speed this up, and the fastest method (at least for very long numbers) involves an unusual application of the Fast Fourier Transform (FFT). As this moves away from the central theme, I will give only the briefest of descriptions here.

This long-multiplication algorithm is a parallel of a manual method in which one forms all of the partial products of single digits, arranging them in columns, and adds them up. In this way, the sums of the columns are analagous to the convolution to two vectors. Now, the FFT of such a convolution is the same as the product of the two individual transforms. So, treating our two numbers as long vectors (like a list of single digits), we calculate the FFT of each, and multiply the corresponding components of the two transforms. Finally, take the inverse FFT to give the product of the two original numbers.

The speed of this algorithm depends on the recursive nature of the FFT, rather than the slower, but equivalent, Discrete Fourier Transform. (For further details of these two, I refer the interested reader to part 5 of this series, in *Archive 23:2*). The two numbers to be multiplied are stored as character arrays, and processed 8 bits at a time. To ensure that no rounding errors creep in, it is necessary to keep sufficient precision when calculating the FFTs, and in this case it turns out that 64-bit floating-point arithmetic is required.

Of course, the devil lies in the detail, namely that, with the exception of the Beagleboard, the Arm processor cannot perform these operations in hardware. To be fair, it must be admitted that, even with hardware floating-point, the algorithm becomes attractive only for very large numbers. However, when necessary, it is a vital component of square-root algorithms.

Rounding up

To summarize, this part of the series has looked at two methods for square roots, both of which have been known long before the advent of electronic computers. Despite this, both are still in regular use in a variety of computer software. Although thay can be sped up, the improvement is small and worthwhile only for very large numbers – a problem that is unlikely to trouble the average Risc os user.

Nicholas Cutler *ncc25@cam.ac.uk*

When this article was completed, Nicholas had returned from a holiday in the Shetland Isles. The journey by ferry from Aberdeen seems long, but then Lerwick is just as close to Bergen!

Risc os contacts

Key: (phone) [fax]

Advantage Six / STD	PO Box 183, Rochdale, OL16 9EW. (01706 84 8600) [0870 164 1604] *www.advantagesix.com info@advantagesix.com*
APDL / ISV	39 Knighton Park Road, Sydenham, London, SE26 5RN. (020 8778 2659) [8488 0487] *www.apdl.co.uk info@*
Castle / Iyonix	*www.iyonix.com sales@* fax [0871 714 3541] *support@* (0797 629 5923, m–f 9–12)
CJE Micros / 4D	78 Brighton Road, Worthing, BN11 2EN. (01903 52 3222) [52 3679] *www.cjemicros.co.uk info@cjemicros.co.uk*
Electronic Font Foundry	101 Lower Village Road, Ascot, SL5 7AP. (01344 87 5201) [87 5202] *www.eff.co.uk fonts@eff.co.uk*
MW Software / Icon Tech	Martin Würthner, Benzinoring 19, 67657 Kaiserslautern, Germany. (+49 631 360 8205) *www.mw-software.com martin@*
Orpheus Internet	(01702) 46 2385 *www.orpheusinternet.co.uk sales@ support@*
David Pilling	Thornton Cleveleys, Blackpool, FY5 1SU–12. **new**: no box n°. [0870 052 0941] *www.davidpilling.net david@pilling.demon.co.uk*
R-Comp / RCI	22 Robert Moffat, High Legh, Knutsford, Cheshire, WA16 6PS. (01925 75 5043) [75 7377] *www.rcomp.co.uk rcomp@rcomp.co.uk*
Riscos Ltd	143 Allensbank Road, Heath, Cardiff, CF14 3PQ. (029 2061 9942) [2021 3156] *www.riscos.com info@*
Risc os Open	4 Townsend Mews, Wilburton, Ely, Cambs, CB6 3SQ. *www.riscosopen.org info@riscosopen.org*
David Snell	*www.dsnell.zynet.co.uk dsnell@mail.zynet.co.uk* (01392 21 4033)
Virtual Acorn / 3QD	29 Alma Rd, Newhall, Swadlincote, DE11 0SD. (01283 52 2969) *www.virtualacorn.co.uk info@virtualacorn.co.uk*
X-Ample Technology	NL-5615 CP Eindhoven. (+31 40 294 0297) *www.xat.nl*

PC Bits: a look thru Windows 8

Bimal Jangra

It seems like an eternity since I last wrote my column in Archive. A lot of changes have happened here at Jangra Towers over the past six months. The most significant is that I finally left the teaching profession once and for all. Things have changed so much since I first became a teacher back in 1981, and I really do not wish to be part of the profession any more – I feel more like a teacher from a bygone age (no, I'm saying that I'm a fogey!) where teachers were allowed to teach instead of feeling that their pupils have more rights than they do. This is a sweeping generalization, of course, but I have had enough negative experiences to convince me that I am far better off being out of school.

Bimal's verdict on Windows 8:

Fine for a touchscreen, not for a desktop

October 26 is the release date for Windows 8 – the eve of the Risc os London show. Microsoft will let existing users of Vista, XP and Windows 7 download the top-level Windows 8 Pro for just $20 until the end of January. The price then jumps to $40 (£25).

Since this article was written, I have gained more experience with Windows 8 and read a great many reviews and articles about it.

Personally, I have no intention of getting a tablet computer. It wouldn't suit the things that I need to do regularly, such as edit photos and manipulate sound (try editing a photo, to get it pixel-perfect using your finger – or drawing a new waveform in a synthesizer!). Windows 8 is definitely aimed first and foremost at tablet users.

The basic interface where you arrive after booting up is designed for tablet users. Several features have been tweaked to give tablets an easier time: there are large icons and different parts of the operating system can be accessed by going to corners or edges, since these are simple movements on a tablet. The system works well – if you are using a tablet; Windows 8 has been rewritten in order to perform well on the trendy hardware that many people have these days.

But what if you are a desktop or laptop user? The story isn't so rosy. Familiar things like the Start menu and taskbar are not immediately accessible – the default environment is aimed at tablets. There isn't space here to cover every detail, but you can easily find all the detail you want online among the endless reviews and comparisons.

The question you want the answer to is – should I get it? Well, if you use a tablet PC, Windows 8 will suit the hardware down to the ground and will give you a powerful system that responds to your touch-driven interface. Get Windows 8 while the prices are low after the launch.

If you use a desktop or laptop computer, I would advise you not to go for Windows 8. If you already have Windows 7, stick with it because it is still very, very good. If you have a system older than Windows 7, upgrade to Windows 7! If you buy a computer with Windows 7, you can get a free upgrade to Windows 8: don't bother! Stick with Windows 7. Especially beware of the difficulty of getting rid of Windows 8 once you find out that you don't *really* like it!

What am I doing now? Well, I was fortunate to secure a position at Capita IT Services, where I work on the Government Service Desk providing technical support for a number of interesting contracts. The job has also given me practical experience of a host of new skills, as well as updating my existing skills – so I now have lots of experience with Active Directory, Citrix environments, the latest Novell systems, and even SAP.

I am loving every minute, and look forward to going to work each day. I consider myself extremely fortunate in the current conditions to have a permanent job that I love doing and enjoy.

Computing developments

The most important event coming up in the PC world is the release of Microsoft's Windows 8, since Microsoft still dominates the operating-system landscape. The behemoth has a few issues to address, however. The biggest problem is that Microsoft in the 21st century actually created some operating systems that have proved just too good! For example, even though support for Windows XP, which came out in 2001, is to end in 2014 (no more updates after then) it is still used by vast numbers of business organizations – most of the clients I currently support still use XP; my machine at work runs XP.

This situation came about because in 2006 when Microsoft set Vista loose on the world, businesses were loath to move from popular XP to very unpopular Vista. Companies found that XP continued to meet their requirements – you can run Microsoft Office 2010 on Windows XP, so why change? "If it ain't broke, don't fix it" became many companies' official stance.

Some did change over to Windows 7 when it was finally released in 2009, and people found that it was what Vista should have been. It loads much more quickly than Vista or XP. Besides that, the Professional

Windows 8's new desktop is even more minimalist than Windows 7. Look what's become of the "Start Orb".

version of Windows 7 (and above, but not Home Premium) comes with XP Mode, which allows you to run a virtual copy of Windows XP Professional SP3: this can run all your favourite XP software.

She's saying Windows 8 will run on Arm-powered devices too.

It integrates so well with the Windows 7 environment that XP Mode software can be listed in your Windows 7 Start menu – when you click on the program name, the virtual XP environment opens and then loads that software. And I have to say it runs very well indeed. It will even run some of my favourite XP-only games (although obviously not any that require particular high-spec hardware or that addresses memory in a non-standard way).

So, if people did move from Windows XP to Windows 7, they will have found that the move was fairly painless and has resulted in an up-to-date system with all of the backwards compatibility they could need. Are they likely to change over to Windows 8?

Change from Windows 7 to 8?

This really is the big question. Microsoft has made no secret that it is aiming at the touchscreen market with Windows 8. The interface has been tweaked to make it easier to use on touchscreens. At the time of writing this I haven't had a proper look at it yet, due to massive constraints on my time. I would very much like to install the pre-release version (available at *windows.microsoft.com/en-US/windows-8/download*) but would install it only into a virtual system such as VMware or Oracle's Virtual Box (it used to be Sun's but this has changed – it's still an awesome free piece of software (from *virtualbox.org/*) and can be run on Windows, Linux, Macintosh and Solaris hosts).

One other detail to bear in mind with Windows 8 is that the pre-release version offers the tester a discounted upgrade once the final product is released (there may be restrictions based on geographical location, though).

A big beware

The other factor that leapt out at me is that installing the pre-release version will destroy the partitioning information of your machine. The instructions warn that if you decide to go back to your previous version of Windows (*"Why would you?" they seem to be asking themselves*), you will need to reinstall it from your original installation media or a full backup – a recovery partition will not be usable. Well, I definitely don't like the sound of that. I never allow software installations to tell me how my drive is to be partitioned. I always want to make the decisions for myself, as I know what I want to have and where.

It would certainly be wise to make a

complete and full backup of your existing system if you decide to try out Windows 8 on your system – unless you have a spare machine and don't care if everything gets overwritten.

Conclusions about Windows 8?

I have no firm conclusions at this point. The articles I read both online and in the computing press seem to suggest that Windows 8 has features that are a natural development on from Windows 7 (which I make no secret about liking a great deal – and I definitely prefer it to Windows XP, much as I still like that) such as the ability to use touch-enabled devices and being fast at loading up initially.

However, despite its intention to be all things to all people, I haven't yet read of anything that makes me think, "Wow, I'm going to have to get Windows 8 as soon as I can." I also don't have any touch-enabled devices, nor am I planning to get any in the near future. What is the point of touch interfaces? Surely they get greasy all the time, and so require continually being cleaned in order to stay crisp and clear? Given the choice of an Iphone or a Blackberry, I would go for the Blackberry (yeah, I know there's a touch-screen Blackberry) as I like having a "proper" keypad for typing text messages and web addresses – I have seen the appalling time people have with touchscreen keyboards! They're still not anywhere near as effective as a real keyboard – and don't even get me started on smart text! What do you think?

That's it for now

As always, if you would like any more information about anything I've written about, please email me, and I will do my best to respond as quickly as I can. And if there are any topics you would like to see covered in this column, please let me know.

Next time, I plan to let you know about a recent Darc Technology Club meeting, where we looked at geotagging and digital forensics. If you're not sure what these terms mean, you'll just have to wait until then! Happy computing!

Bimal Jangra *hermits@btinternet.com*

Bimal has been editing *The Spotlight*, a 44-page magazine, since January. The magazine is produced on behalf of the Midlands Counties Entertainment Secretaries Council and includes reports on the artists performing in the clubs. He has given it a complete makeover and is delighted with the positive responses from everyone.

The cover is designed in Xara Designer Pro 7 (he'll get 8 as soon as finances permit) and the content is laid out in Serif Page Plus X6. The result is sent to the printshop as PDF.

Midland Counties Entertainment
July 2012
THE SPOTLIGHT
Your Guide to Entertainment in the Midlands' Clubs
Find us on Facebook
Forthcoming Shop Windows:
Sunday 1st July 12 noon @ Pensnett & Bromley Memorial Club
Sunday 5th August 12 noon @ Baggeridge Social Club
Call those MASCOTS?!
60p

eXtreme eXperimental eXperience eXciting eXcellent eXhilarating ARMiniX

The ARMiniX

What does the X mean for you?

- ARM Cortex A9 at 1.2 GHz
- 1 gigabyte ram
- 32GB high-speed flash storage
- DVD read-write drive
- disc operations 4× Armini
- RISCOS 5.2?

Software bundle includes:

- Aemulor Pro
- !MusicCD

SEE IT AT THE RISCOS LONDON SHOW and check out the new !store

!Pilingstore — the RISC OS application store

RComp Interactive

www.armini.co.uk

phone 01925 75 5043